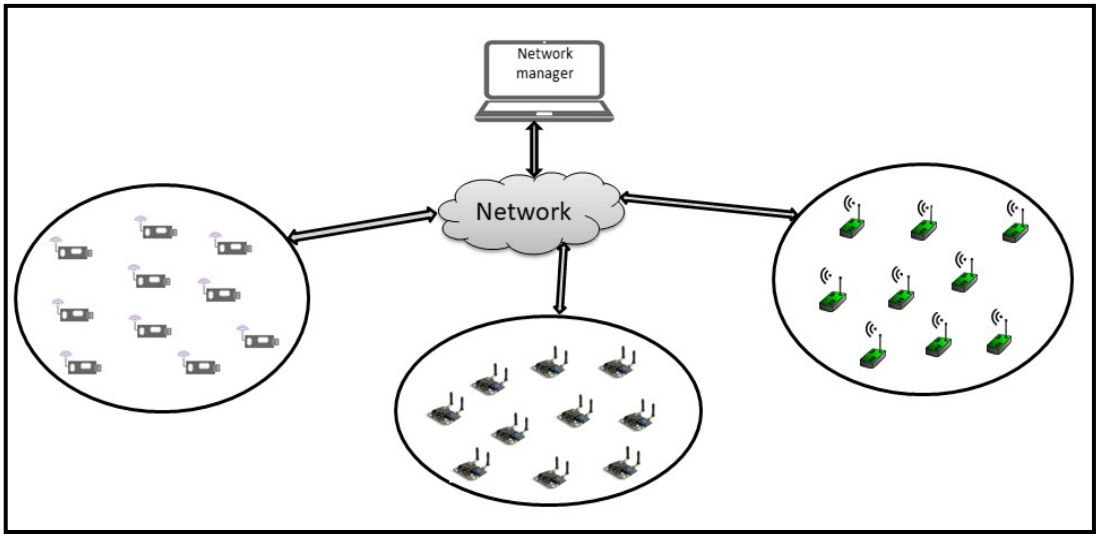
Efficient management of IoT low power networks

Moussa Aboubakar

Par **Moussa ABOUBAKAR**

Efficient management of IoT low power networks

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 11 décembre 2020
Spécialité : Informatique et Sciences et Technologies de l'Information et des Systèmes : Unité de recherche Heudyasic (UMR-7253)
D2571

Efficient management of IoT low power networks

THESE DE DOCTORAT

Soutenue le 11 décembre 2020

pour l'obtention du

DOCTORAT de l'Université de Technologie de Compiègne

Spécialité: Informatique et Sciences et Technologies de l'Information et des Systèmes

par **Moussa ABOUBAKAR**

Membres du jury:

Président du jury:	Pr. Walter SCHÖN	Professeur, UTC
Rapporteurs:	Pr. Lorenz PASCAL	Professeur, IUT de Colmar
	Pr. Ahmed MEHAOUA	Professeur, Université de Paris Descartes
Examinatrice:	Dr. Kervella BRIGITTE	Maître de Conférences, UPJV
Encadrant:	Dr. Mounir KELLIL	Ingénieur chercheur, CEA
Invité	M. Pierre ROUX	Ingénieur chercheur, CEA
Directeur de thèse:	Pr. Abdelmadjid BOUABDALLAH	Professeur, UTC

Quote

"When you want something, all the universe conspires in helping you to achieve it."

Paulo coelho

Abstract

In these recent years, several connected objects such as computer, sensors and smart watches became part of modern living and form the Internet of Things (IoT). The basic idea of IoT is to enable interaction among connected objects in order to achieve a desirable goal. IoT paradigm spans across many areas of our daily life such as smart transportation, smart city, smart agriculture, smart factory and so forth. Nowadays, IoT networks are characterized by the presence of billions of heterogeneous embedded devices with limited resources (e.g. limited memory, battery, CPU and bandwidth) deployed to enable various IoT applications. However, due to both resource constraints and the heterogeneity of IoT devices, IoT networks are facing with various problems (e.g. link quality deterioration, node failure, network congestion, etc.). Considering that, it is therefore important to perform an efficient management of IoT low power networks in order to ensure good performance of those networks. To achieve this, the network management solution should be able to perform self-configuration of devices to cope with the complexity introduced by current IoT networks (due to the increasing number of IoT devices and the dynamic nature of IoT networks). Moreover, the network management should provide a mechanism to deal with the heterogeneity of the IoT ecosystem and it should also be energy efficient in order to prolong the operational time of IoT devices in case they are using batteries. Thereby, in this thesis we addressed the problem of configuration of IoT low power networks by proposing efficient solutions that help to optimize the performance of IoT networks. We started by providing a comparative analysis of existing solutions for the management of IoT low power networks. Then we propose an intelligent solution that uses a deep neural network model to determine the efficient transmission power of RPL networks. The performance evaluation shows that the proposed solution enables the configuration of the transmission range that allows a reduction of the network energy consumption while maintaining the network connectivity. Besides, we also propose an efficient and adaptative solution for configuring the IEEE 802.15.4 MAC parameters of devices in dynamic IoT low power networks. Simulation results show that our proposal improves the end-to-end delay compared to the usage of the standard IEEE 802.15.4 MAC. Additionally, we develop a study on solutions for congestion control in IoT low power networks and

propose a novel scheme for collecting the congestion state of devices in a given routing path of an IoT network so as to enable an efficient mitigation of the congestion by the network manager (the device in charge of configuration of the IoT network).

Keywords : *Internet of Things, Low power and lossy networks, network management, network performance, machine learning.*

Resumé

Durant cette dernière décennie, plusieurs objets connectés tels que les ordinateurs, les capteurs et les montres intelligentes ont intégrés notre quotidien et forment aujourd'hui ce que l'on appelle l'Internet des Objets (IdO) ou Internet of Things (IoT) en anglais. L'IoT est un nouveau paradigme permettant une interaction entre les objets connectés afin d'améliorer notre qualité de vie, notre façon de produire des biens et notre façon d'interagir avec notre environnement. De nos jours, l'IoT se caractérise par la présence, de par le monde, de milliards d'objets connectés à faibles ressources (batterie, mémoire, CPU, bande passante disponible, etc) et hétérogènes, déployés pour permettre diverses applications couvrant de nombreux domaines de notre société tels que la santé, l'industrie, les transports, l'agriculture, etc. Cependant, en raison des contraintes lié aux ressources et de l'hétérogénéité des objets connectés, les réseaux IoT à faibles ressources présents font face à des problèmes de performance, notamment la dégradation de la qualité des liens radio, la défaillance (logiciele ou matérielle) de certains objets du réseau, la congestion du réseau, etc. Ainsi, il est donc important de gérer efficacement les réseaux IoT à faible ressources afin d'assurer leur bon fonctionnement. Pour ce faire, la solution de gestion du réseau doit être autonome (pour faire face à la nature dynamique des réseaux IoT), tenir compte de l'hétérogénéité des objets connectés et être moins consommatrice en énergie pour répondre aux défis de l'IoT. Dans cette thèse, nous nous sommes intéressés au problème de gestion des réseaux IoT à faibles ressources et avons proposés des solutions efficaces pour permettre une optimisation des performances de ces types de réseaux. Dans un premier temps nous avons procédé à une étude comparative des solutions de gestion des réseaux IoT à faibles ressources afin d'identifier les verrous techniques. Ensuite, nous avons proposé une solution intelligente qui se base sur un modèle de réseau de neurones profonds pour permettre une configuration de la portée radio dans les réseaux sans fil à faibles ressources de type RPL (IPv6 Routing Protocol for Low power and Lossy Networks). Une évaluation des performances de cette solution montre qu'elle est capable de déterminer la portée radio permettant une réduction de la consommation énergétique du réseau tout en garantissant une connectivité des objets connectés. Nous avons également proposé une solution efficace et adaptative pour configurer les paramètres de la couche MAC

dans les réseaux dynamiques de type IEEE 802.15.4. Les résultats des simulations démontrent que notre solution améliore le délai de transmission bout en bout par rapport à l'utilisation des paramètres par défaut de la MAC IEEE 802.15.4. En outre, nous avons proposé une étude des solutions existante pour la gestion des problèmes de congestion des réseaux IoT à faibles ressources et par la suite nous avons proposé un procédé d'acheminement de l'information de congestion des objets connectés présents sur un chemin de routage donné dans des réseaux à ressources limitées. Cette méthode a pour but de permettre une réponse efficace aux problèmes de congestion.

Mots Clés : *Internet des Objets, gestion des réseaux, performance des réseaux, machine learning.*

Acknowledgment

First of all, I would like to thank God for all his blessing and for me giving strengh to achieve this work.

I would like to express my sincere gratitude to my thesis supervisors professor Abdelmadjid BOUABDALLAH and Dr Mounir KELLIL. They provided me with valuable advice, encouragement and guidance that helped me to carry out this research work.

I would like to express my deep gratitude to M. Pierre ROUX for his availability since the beginning of this thesis and also for fruitful discussions and valuable remarks that helped in better improving the work done during this research work.

I would like to thank Pr. Lorenz PASCAL and Pr. Ahmed MEHAOUA for their valuable time in reviewing this manuscript. I would also like to thank Dr. Kervella BRIGITTE and Pr. Walter SCHÖN for their precious work as examiners.

I would like to thank Dr Ari Ado ABBA ADAMOU for his advice that helped me strengthening my motivation for this PhD work.

I would like to thank all my colleagues in LSC and Heudiasyc laboratories for their sympathy and all good times we spent together.

Last but not the least, I would like to thank my family, especially my father Moussa ABOUBAKAR and my mother KOAH ABONE Yvonne Seraphine for supporting me during my whole life.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

AdaR:	Adaptive Routing
ASCI:	Agriculture Sensor-Cloud Infrastructure
AGR:	Aggregation Gradient Routing
AMQP:	Advanced Message Queuing Protocol
BLE:	Bluetooth low Energy
BE:	Backoff exponent
CSMA-CA:	Carrier Sense Multiple Access with Collision Avoidance
CMIP:	Common Management Information Protocol
CEB:	Cloud-Edge-Beneath
CEA:	French Alternative Energies and Atomic Energy Commission
CIB:	Congestion Information Block
CNN:	Convolutional neural network
CODA:	Congestion Detection and Avoidance
CoMI:	CoAP Management Interface
CoAP:	Constrained Application Protocol
CC:	Correlation Coefficient
CW:	Contention Windows
DAO:	Destination Advertisement Object
DDS:	Data-Distribution Service for Real-Time Systems
DBN:	Deep Belief Network
DIS:	DODAG Information Solicitation
DIO:	DODAG Information Object
DODAG:	Destination Oriented Directed Acyclic Graph
DIRL:	Distributed Independent Reinforcement Learning
DRL:	Deep-Reinforcement Learning
ECODA:	Enhanced Congestion Detection and Avoidance
ECN:	Explicit Congestion Notification
ETX:	Expected Transmission Count
ELT:	Expected Lifetime metric

EDRb:	Energy Distance Ratio per bit
FACC:	Fairness-Aware Congestion Control
FROMS:	Feedback ROuting to Multiple Sinks
H-SMSR:	Hierarchical Scalable Multipath Source Routing
HSR:	Hierarchical Source Routing
IBA:	Index of Balanced Accuracy
ICN:	Implicit Congestion Notification
IoT:	Internet of Things
IdO:	Internet des Objets
IDS:	Intrusion Detection System
IWSN:	Intelligent Wireless Sensor Network
IETF:	Internet Engineering Task Force
KNN:	K-Nearest Neighbor
LSPI:	Least Squares Policy Iteration
LPWA:	Low Power Wide Area
LR-WPAN:	Low-Rate Wireless Personal Area Network
MAC:	Media Access Control
MAE:	Mean Absolute Error
MSE:	Mean Square Error
MIB:	Management Information Base
ML:	Machine Learning
MDP:	Markov Decision Process
MQTT:	Message Queuing Telemetry Transport
MQTT-SN:	MQTT for Sensor Networks
MLP:	Multi-layer Perceptron
NETCONF:	Network Configuration Protocol
OSI:	Open system Interconnect
OFs:	Objective Functions
OMA-DM:	Open Mobile Alliance - Device Management
PRR:	Packet Reception Rate
QoS:	Quality of Service
RMSE:	Root Mean Square Error
RNN:	Recurrent Neural Network
RPL:	IPv6 Routing Protocol for Low-Power and Lossy Networks
SDN:	Software Defined Network
SI:	Stability Index
SDCSN:	Software-Defined Clustered Sensor networks

SVM:	Support Vector Machine
SVMs:	Support Vector Machines
SNMP:	Simple Network Management Protocol
SMI:	Structure of Management Information
TPE:	Tree-structured Parzen Estimators
UHCC:	Upstream Hop-by-hop Congestion Control
WSN:	Wireless Sensor Network
WSNs:	Wireless Sensor Networks
XMPP:	Extensible Messaging and Presence Protocol

List of Publications

1 International Conference

- ◇ **Moussa Aboubakar**, Mounir Kellil, Abdelmadjid Bouabdallah, Pierre Roux. "Toward intelligent reconfiguration of RPL networks using supervised learning." *In proceedings of 2019 IEEE Wireless Days (WD), April 2019, pages 1-4.*
- ◇ **Moussa Aboubakar**, Mounir Kellil, Abdelmadjid Bouabdallah, Pierre Roux. "Using Machine Learning to Estimate the Optimal Transmission Range for RPL Networks." *In Proceedings of NOMS 2020 IEEE/IFIP Network Operations and Management Symposium, April 20, pp. 1-5.*
- ◇ **Moussa Aboubakar**, Pierre Roux, Mounir Kellil, Abdelmadjid Bouabdallah. "An Efficient and Adaptive Configuration of IEEE 802.15.4 MAC for Communication Delay Optimisation" *In Proceedings of 11th International Conference on Network of the Future, October 2020.*
- ◇ **Moussa Aboubakar**, Pierre Roux, Mounir Kellil, Abdelmadjid Bouabdallah. "CIB: Congestion Information Block for IoT low power networks." *Submitted to 17th IFIP/IEEE Symposium on Integrated Network and Service Management (IM 2021)* .

2 International Journal

- ◇ **Moussa Aboubakar**, Mounir Kellil, Pierre Roux and Abdelmadjid Bouabdallah "Machine Learning for Congestion Control in Wireless Sensor Networks: A comparative analysis" *Submitted to Springer Journal anals of telecommunications.*
- ◇ **Moussa Aboubakar**, Mounir Kellil and Pierre Roux "A review of IoT Network Management: Current Status and Perspectives" *Submitted to Elsevier Journal of King Saud University-Computer and Information Sciences.*

3 Patent

- ◇ **Moussa Aboubakar**, Mounir Kellil, Pierre Roux. "Procédé de transmission d'information de gestion dans un réseau maillé" *Patent submitted in july 2019 to INPI.*

4 Other Publications

The others research activities conducted by the author are given in the following

- ◇ Blaise Omer Yenké, **Moussa Aboubakar**, Chafiq Titouna, Ado Adamou Abba Ari, Abdelhak Mourad Gueroui. "Adaptive scheme for outliers detection in wireless sensor networks." *International Journal of Computer Networks and Communications Security 5.5 (2017): 105.*

General Introduction

Context

The Internet of Things (IoT) has witnessed a rapid growth these last years thanks to the development of smart devices and new communication technologies. Coined by Kevin Ashton [17], the term IoT refers to a paradigm where the physical objects of our daily life (e.g. sensors, actuators, home appliances and so forth) are connected to the internet and are able to communicate in an intelligent fashion. Nowadays, IoT landscape encompasses billions of heterogeneous devices which enable a wide range of applications that improve our daily life. These applications include smart homes, smart cities, smart agriculture, factory of the future and so forth.

In the current IoT ecoystem, embedded devices with limited power, memory, and processing resources are enablers of an important number of IoT applications. This has been made possible by the development of different protocols [140] (e.g. RPL , CoAP, 6LoWPAN, etc.) and communication technologies [170] (e.g. BLE, Zigbee, NFC, etc.). Nevertheless, due to resource limitations of IoT devices and often the constraints related to the deployment of those devices in hard to reach areas (e.g. pipelines, war zones, earth-quake or chemical spill areas), IoT networks are facing many problems that affect their performance. These problems include link quality deterioration, network congestion, failure of devices, and contribute to a significant reduction of the performance of IoT networks. Besides, it is worth to note that IoT networks are suffering from heterogeneity of the IoT ecosystem and the scalability issue due to the increasing number of IoT devices and/or the increasing number of IoT applications. In this context, it is therefore important to perform an efficient management of IoT networks in order to ensure good performances and to maintain the network operational for a longest period of time possible when battery powered devices are used [172].

In the literature, various network management protocols have been proposed by different standardization organizations such as Internet Engineering Task Force (IETF), OneM2M and Broadband Forum [133, 125, 121], to help to perform configuration of IoT devices. Likewise, different solutions for management of IoT low power networks have been proposed by the research community [110, 175, 160].

Those protocols and solutions for network management can operate in a centralized, distributed or hybrid architecture.

Since IoT networks are being exponentially deployed both in public (smart cities, smart buildings, etc.) and private areas (smart homes, smart factories, etc.), network managment becomes the cornerstone of IoT networks to achieve the best network performance and maintain a high level of network availability. Nevertheless, achieving such a performance objective is not straightforward, because of the intrinsic characteristics of IoT low power networks (scarce resources, lossy channels, device/node failure, and device/node deployment constraints like node/device reachability). In view of that, the network management solutions need to fulfill a number of critical features, encompassing scalability, self-configuration, and energy-efficiency. However, these features are not well considered by the existing network management solutions [138, 141].

Objectives and contributions

The objective of this thesis is to address the problem of management of IoT low power networks and propose efficient solutions to those problems. Our contributions are summarized in what follows:

1) In our first contribution, we provided a survey that includes a classification of existing solutions for IoT network management and proposes a comparative study of those solutions according to different requirements. Moreover, we pointed out a number of challenges and open issues relating to IoT networks management.

2) In our second contribution, we proposed centralized solutions that enable to configure the transmission power of IoT low power networks based on a particular machine learning model called Multilayer Perceptron (MLP). The proposed solutions enable a reconfiguration of dynamic RPL networks deployed following a 2D or a 3D environment. We performed simulation of these solutions and compared them with other machine learning models. We evaluated our solutions based on various metrics such as the accuracy of the machine learning models and the energy consumption. The results show that our solutions can help to define the transmission power that enables to reduce the network energy consumption while maintaining the network connectivity.

3) In our third contribution, we proposed an intelligent solution that enables to configure the IEEE 802.15.4 MAC parameters for dynamic IoT low

power networks based on machine learning techniques such as multiouput decision tree, multioutput random forest, multioutput K-nearest neighbors and multioutput MLP. The proposed solution enables the selection of the optimal IEEE 802.15.4 MAC parameters according to the network characteristics and network traffic conditions. We performed and evaluation of the proposed solution based on metrics such as end-to-end delay and the accuracy of the machine learning model. The results obtained prove that the MAC parameters estimated using the random forest classifier perform better compared to the default MAC parameters of IEEE 802.15.4 standard.

4) In our fourth contribution, we proposed a comparative analysis of a number of solutions for congestion management in resource-constrained networks. We highlighted the need for a machine learning solution for congestion control in resource-constrained networks. We provided a guide for devising machine learning based solution with imbalanced data for congestion control in resource-constrained networks.

5) In our fifth contribution, we proposed a novel scheme for efficient agregation of congestion information in IoT low power networks. The proposed scheme enables the transmission of congestion state of nodes in given routing path into a single data packet sent to the network manager. The congestion states are aggregated into a block called Congestion Information Block (CIB), which contains binary values representing the state of nodes. The simulation results obtained show that the proposed scheme provides a good performance compared to Explicit Congestion Notification (ECN) mechanism in terms of network throughput, network overhead and offers low divergence (regarding the time of observation) between the network congestion observed by the network manager and the real congestion of nodes.

Organization of the manuscript

The rest of this thesis is organized as follows.

- In Chapter 1, we give a background knowledge concerning IoT low power networks. We firstly provide an overview on IoT low power networks, followed by a presentation of the architecture of IoT networks. Then, we present some applications of IoT low power networks and provide a presentation of requirements and challenges of IoT low power networks.

- In Chapter 2, we present a comprehensive overview on existing solutions for IoT networks management. In particular, we provide a classification of existing solutions for management of IoT low power networks according to their design objectives. Then, we provide a comparative analysis of those solutions according to different requirements.

- In Chapter 3, we present the proposed intelligent solution that determine the efficient transmission power of devices in IoT low power networks. This solution uses a deep neural network model to estimate the efficient transmission power of an RPL network. We firstly provide an overview on existing methods for configuring the transmission power of IoT low power networks. Then, we provide a description of the proposed models that enable to determine the efficient transmission power of static and dynamic IoT low power networks. We also provide the obtained performance evaluation results and a description of deployability of the proposed solution.

- In Chapter 4, we present the proposed solution that helps to efficiently configure the IEEE 802.15.4 MAC parameters in dynamic IoT low power networks. Our solution is based on a supervised learning technique and it enables to define the IEEE 802.15.4 MAC parameters according to the current network traffic conditions and the network characteristics. We start this chapter by presenting background knowledge regarding the IEEE 802.15.4 MAC. Then, we present an overview of existing methods for configuring the IEEE 802.15.4 MAC parameters. After that, we present the proposed solution that allows an efficient configuration of IEEE 802.15.4 MAC parameters and we evaluate of the performance of our proposed solution.

- In Chapter 5, we present a comparative analysis of existing solutions for congestion control schemes in resource-constrained networks. Then, we point out the need for a machine learning based solution for congestion control in resource-constrained networks. Moreover, we provide a guide for applying machine learning techniques for congestion control in resource-constrained networks based on imbalanced data.

- In Chapter 6, we present the proposed scheme for efficient aggregation of congestion state of nodes, in a given routing path, into a block called Congestion Information Block (CIB). This proposal enable an efficient notification of congestion state of IoT devices managed by a central entity. We confirm the efficiency of our proposal through series of simulation.

- In Chapter 7, we conclude this research work by presenting a summary of our contributions and present a number of research perspectives.

Part I

Literature review

Background

Sommaire

1.1 Introduction

For more than a decade, IoT networks have been widely investigated by reseachers due to the potential applications of those networks in our everyday's life (e.g., smart cities, smart agriculture, health monitoring, etc.). IoT networks are composed of a large number of devices that collaborate together in order to provide a service for the benefit of our society. In particular, among those devices, billion of them are resource-constrained and they constitute IoT low power networks. In this chapter, we provide a background knowledge about those networks.

The rest of the chapter is organized as follows: Section 1.2 presents an overview on IoT low power networks. Section 1.3 presents the architecture of IoT networks. Section 1.4 presents applications of IoT low power networks. Section 1.5 lists different requirements of IoT low power networks. Section 1.6 concludes this chapter.

1.2 Overview of IoT low power networks

In the current ecosystem of Internet of Things (IoT), IoT low power networks have attracted a lot of attention from industrials and academics due to their potential

applications in our daily life. An IoT low power network refers to a network composed of devices (e.g. sensors, actuators) with limited resources (memory, battery, bandwidth and CPU) and interconnected using lossy links. In the literatue, we found that the are two main categories of IoT low power networks [9]: 1) Low Power Wide Area Networks (LPWA) and 2) Low-Rate Wireless Personal Area Network (LR-WPAN). We provide in Table 1.1, a description of those networks.

Table 1.1 – Different IoT low power networks.

Network	**Proprietary technologies**	**Standards**	**remarks**
Low Power Wide Area (LPWA) Networks	Sigfox, LoRa, Ingenu RPMA, Telensa	LoRaWAN, enhanced MTC (eMTC), NarrowBand IoT (NB-IoT), 802.15.4g, DASH7	Supports long range transmission and low data rate transmission
Low-Rate Wireless Personal Area Network (LR-WPAN)	-	6LowPAN, Zigbee, Bluetooth low Energy (BLE), Z-Wave, WirelessHart	Supports short range transmission and low rate data transmission

1.3 Architecture of IoT networks

In the literature, different architecture for IoT have been proposed [134, 96]. These architectures include three or five layer architectures as shown in Figure 1.1. The three-layers architecture represents the basic architecture of IoT while the five-layers architecture represents an extension of the three layer architecture to meet the expected development of IoT.

- ***Perception layer:***
  The perception layer represents the physical layer and it enables an interaction with physical devices and components through smart devices (e.g. RFID, sensors, actuators and so forth). This layer aims at sensing some physical parameters or identifying other smart objects in the environment.

- ***Network layer:***
  The network layer is used to determine the route for information provided

by the perception layer. By doing so, it enables the interconnection of IoT devices.

- ***Application layer:***
  The application layer provide various application service to users. These applications include smart agriculture, smart transportation, smart cities and so forth.

- ***Transport layer:***
  The transport layer layer is responsible for transfering data from the perception layer to the processing by using networks such as BLE, RFID, etc.

- ***Processing layer:***
  The processing layer is responsible for storing, analyzing and processing data received from the transport layer. This layer can manage and provide a set of services to the lower layers. For example, in smart transportation, this layer can provide a prediction of the future traffic conditions by processing the data about the traffic information (e.g. traffic flow, occupancy rate, etc.).

- ***Business layer:***
  The business layer is responsible for the management of the entire IoT system, including applications, business and profit models and users privacy.

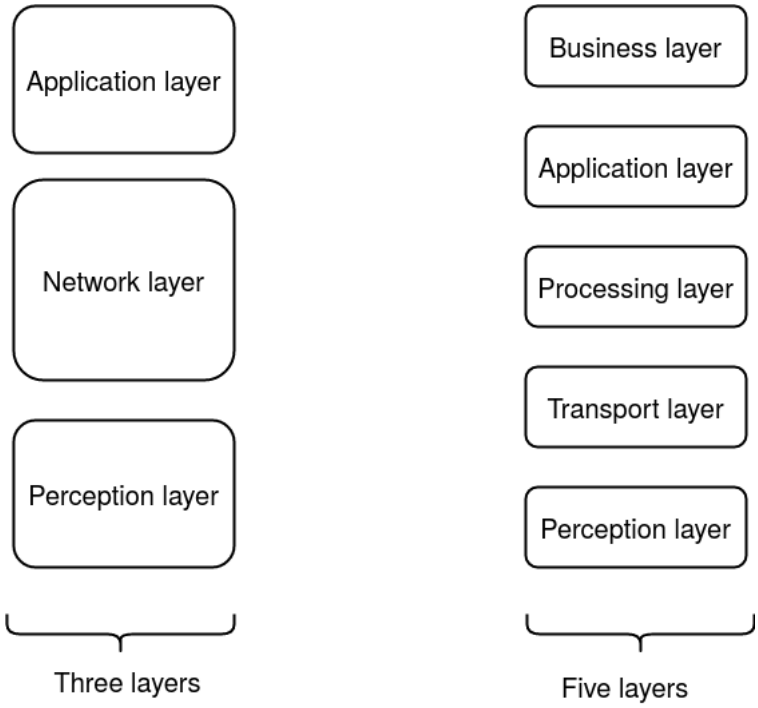


Figure 1.1 – IoT architectures.

1.4 Applications of IoT low power networks

Due to capabilities offered by IoT low power devices, including sensing, actuation, and communication, various applications have emerged in our daily lives. These applications include smart home, smart factory, smart agriculture and so forth. We provide an overview of those applications in Figure 1.2

- ***Smart home***:
  Smart home refers to a house equipped with devices such as sensors or other appliances (e.g. refrigerators, laundry machines) that are connected to IoT in order to control features (e.g. lighting) of the home. The interest for such houses lies in the fact that they allow remote control of the devices in house and enable an efficient management of the energy consumption of those devices.

- ***Smart city***:
  Smart city corresponds to a city that uses various devices such as sensors or actuators in order to collect data (e.g. from people or from environment), analyze them and take a decision that helps to improve our daily life. The benefits of a smart city can be observed in many use cases, notably water supply distribution, waste management, parking management, traffic management, environment monitoring and so forth. To date, many cities accross the world are shifting from traditional city towards smart city [12].

- ***Smart factory***:
  A smart factory is a production facility composed of sensors and other connected devices that aim at improving the industrial processes through automation (to reduce human error in production line) and self-optimization. Concretely, smart factories foster the increase of the productivity and the reduction of the costs of production.

- ***Smart agriculture***:
  Smart agriculture refers to using IoT devices such as sensors in farming to enhance the agricultural production and reduce the cost of production. In fact, those devices can help to monitor parameters such as soil and water quality in order to provide farmers with a mean to adapt their strategy (e.g. adjustment of the quantity of pesticides).

- ***Smart healthcare***:
  Smart healthcare corresponds to health system where IoT devices such as sensors and wearable devices are used in order to improve the diagnosis of a disease and improve the treatment of the patients. The benefits of such

system for a doctor is the improvement of its efficiency by accessing to more information about patients (e.g. through real time monitoring of patients), which in turn will be satisfied by the outcomes of their treatments.

- ***Smart transportation***:
  Smart transportation refers to a transport system based on modern technologies and management strategies. In such transport systems, IoT devices such as sensors can be used in vehicles for collision avoidance and anti-skidding to increase the safety of the system. Moreover, those systems can help to reduce traffic congestion and to improve the quality of the environment.

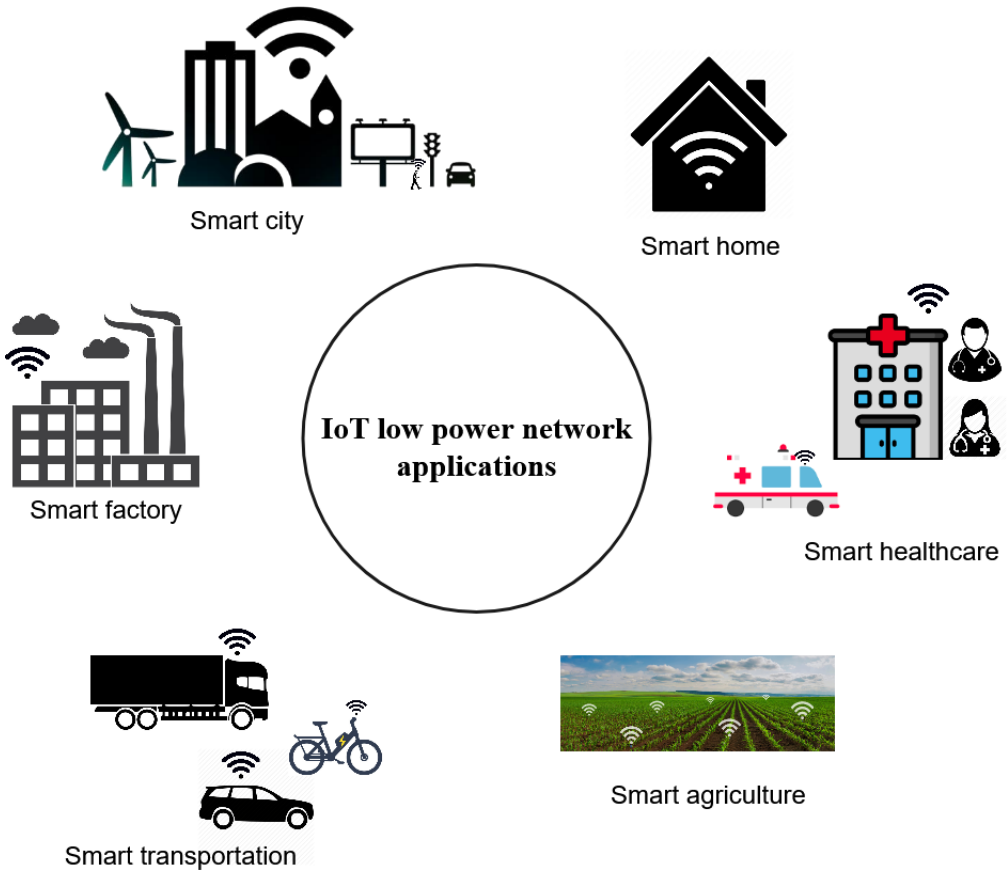


Figure 1.2 – Overview of IoT low power networks applications.

1.5 Requirements and challenges of IoT low power networks

1.5.1 Requirements

In order to operate under a good performance, IoT low power networks need to satisfy requirements as: scalability, fault tolerance, energy efficiency, Quality of Service (QoS) and security.

- ***Scalability***
  A scalable IoT low power network corresponds to a network where new devices or services can be added without negatively affecting the network performance. As the current deployment of IoT low power networks low power is characterized by the presence of billion of resource-constrained devices, the scalability requirement need to be satisfy in order to avoid having poor network performance.

- ***Fault tolerance***
  Fault tolerance is the ability of a system to continue operating in the event of failure of any of its components [35]. This requirement is necessary in order to guarantee that the network will fulfill its expected functioning in presence of fault (e.g. node fault, network fault, sink fault, software fault) in the network. In particular, this requirement is important for IoT low power networks since they may be subject to failure of devices because of their characteristics (limited battery, memory and CPU) and/or the environment in which they are deployed (e.g. war zone, pipeline, chemical spill area).

- ***Quality of Service (QoS)***
  QoS refers to the measurement of overall performance of service in order to assess user satisfaction. This performance is evaluated using these metrics: packet loss, latency, bandwidth and end-to-end delay in the network. Concretely, the level of QoS in IoT low power networks depends on the type of application. For example, IoT applications such as smart metering are delay tolerant while another IoT applications (e.g. forest fire detection) are not delay tolerant. Therefore, to avoid having poor network performance, it is important to consider the QoS requirement when designing the network.

- ***Energy effeciency***
  One of the main requirement of IoT low power is the energy efficiency [126]. An energy efficient network is a network that has the capability to execute

operations with a minimum energy consumption so that the network lifetime can be maximized. This requirement is particularly desirable in an IoT low power network since its composed of devices powered with battery which has a limited lifetime and often cannot be replace. Moreover, if the energy of the resource-constrained devices is consumed quickly, the network may experience a loss of connectivity which may cause an interruption of the network.

- ***Security***
  Security is an important concerns for IoT network as reviewed in [61]. In fact, having a secure network may help to prevent the potential risks for tampering the communication data by unauthorized entity. Having a secure IoT low power network is particularly desirable to ensure the security of data exchanged by the different devices involved in the network. Nevertheless, in IoT low power networks, more attention should be paid because the mechanisms for security developed for traditional network are not always suitable for resource-constrained devices [91].

- ***Self configuration***
  This requirement refers to the capability of IoT low devices to adapt their behavior according to the network state. In fact, self configuration is important for IoT low power networks since these networks are subject to frequent update caused by the traffic patterns, the mobility of devices, the failure of devices and so forth. Moreover, this requirement is necessary because it is not realistic to perform manual configuration of billion of IoT low power devices. Thereby, having a self configurable IoT low power network can help to avoid human error due to manual configuration, and thus help to provide good network performances.

1.5.2 Challenges

Fulfilling the requirements of IoT low power networks mentioned in the previous section remains a research challenge due to the limitation of resources of IoT devices, the dynamic nature of IoT low power networks and the constraints related to the environment in which IoT devices may be deployed (e.g. war zone, pipeline, etc.). In the current literature, various network management protocols, network management platforms and various mechanisms have been proposed in order to meet those requirements. However, the proposed solutions are still not able to address correctly the requirements of IoT low power networks. This is because they are not suitable for resource-constrained devices and they do not address well the heterogeneity of

IoT devices [79, 141]. Therefore, in order to satisfy the requirements of IoT low power networks, it is necessary to develop mechanisms or protocols that help to realize an efficient management of resource-constrained devices [172].

1.6 Conclusions

In this chapter, we provided basic knowledge concerning IoT low power networks. In addition, we provided the requirement of those networks and pointed out the need for having solutions for efficient management of IoT networks so as to ensure good network performance. In the next chapter, we will dive into existing solutions for management of IoT low power networks.

Management of IoT low power networks

Sommaire

2.1 Introduction

In order to ensure a good functioning of IoT low power networks, different network management solutions have been proposed in the literature [125, 110, 11, 83, 14]. These solutions are ranging from lightweight networks management protocols, SDN-based frameworks, cloud-based frameworks, Semantic-based frameworks and Machine Learning based frameworks. In fact, these solutions have been developed in order to meet the requirement of IoT low power networks mentioned earlier (cf. section 1.5) and to cope with the heterogeneity of IoT ecosystem (e.g. in terms of IoT device protocols) and the inherent resource constraints of IoT devices (limitation in term of battery, memory, CPU, bandwidth and so forth). In this chapter, we present a comprehensive review on existing management solutions of IoT low power networks.

The rest of the chapter is organized as follows: section 2.2 presents an overview on traditional network management; section 2.3 presents different frameworks for management of IoT low power networks; section 2.4 concludes this chapter.

2.2 Overview of traditional network management

Network management consists in performing operations such as devices monitoring, routing management and security management in order to ensure a good network performance (e.g. low latency, low energy consumption, low packet loss, etc.). Basically, a typical network management is based on three logical elements: network manager, managed devices and agents. Figure 2.1 gives an overview of the different functional elements involved in network management. The "network manager" represents the device used to manage a group of managed nodes. A "Managed device" refers to a network device exposing a number of parameters (e.g. IP address, CPU usage, residual battery, etc.) that are managed (through read/write operations) by the network manager. The "agent" refers to the software which runs on managed device. It collects raw data from the managed device to transfer it, in a comprehensible or exploitable format, to the network manager. The "management database" contains information concerning the managed device parameters. The "messaging protocols" can be used to exchange information between the network manager and the managed devices. This allows the network manager to get parameters from managed devices and accordingly take appropriate decision concerning the reconfiguration of network devices.

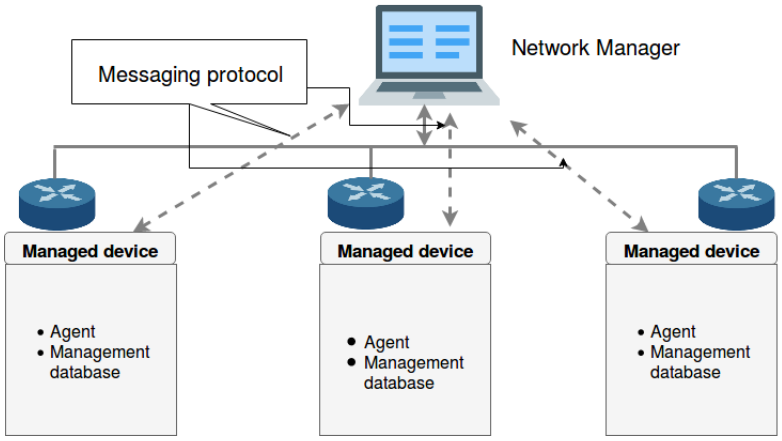


Figure 2.1 – Network management entities overview.

Typically, a network management system needs to support the following operations:

1) **Configuration management**: It refers to the process that helps setting devices parameters such as IP adress and routing table. It encompasses different operations related to the configuration and reconfiguration of all the (writable) network device parameters.

2) **Topology management:** It corresponds to a set of operations that help to maintain the network connectivity while providing good network performance.

3) **Security management:** This operation prevents unauthorized access to an intruder. For this purpose, it includes a wide range of operations such as encryption (key distribution techniques), threat detection and recovery.

4) **QoS management:** It refers to a mechanism that helps to configure the network so as to obtain a desirable network performance in term of data latency, packet loss, throughput.

5) **Fault management:** It corresponds to a mechanism that helps detecting, isolating and resolving network problems without affecting the proper functioning of the network.

6) **Network maintenance:** It refers to a set of operations to perform in order to maintain the network running. It encompasses operations such as software maintenance (e.g. firmware update and bug fixes) and troubleshooting network problems.

To manage traditional networks, various network management protocol such as SNMP [103], CMIP [70], NETCONF [47], RESTCONF [164], CWMP [121] and OMA-DM [13] have been proposed.

- ***Simple Network Management Protocol (SNMP)***
  SNMP is a network protocol developed by IETF (Internet Engineering Task Force) for remote monitoring of IP devices. It supports a set of operations including monitoring, configuration and/or reconfiguration of network device parameters. SNMP involves the three elements of network devices management (agents, nodes and manager) described above. It relies on Structure of Management Information (SMI) and Management Information Base (MIB). MIB designates database used for managing the entities in communication network while SMI defines the structure and types of objects stored in the MIB.

- ***Common Management Information Protocol (CMIP)***
  CMIP is a network protocol responsible for the communication between the network manager and the managed devices. CMIP enables various network management operations such as fault management, security management, performance monitoring and so forth. CMIP was designed to be used on Open Systems Interconnection (OSI) and it extends the capability of SNMP.

Nevertheless CMIP has not been widely adopted because of slowness in the process of its standardization.

- ***Network Configuration Protocol (NETCONF)***
  NETCONF has been introduced to improve SNMP. It introduces new features in network management such as multiple configuration data stores (candidate, running, startup), distinction between configuration and state data. NETCONF uses Extensible Markup Language (XML) based data encoding for both the configuration data and the protocol messages. NETCONF uses the YANG model which is a data modeling language used to model configuration and state data manipulated by the NETCONF protocol [25].

- ***RESTCONF***
  RESTCONF protocol has been designed with the goal of extending NETCONF protocol in order to enable the possibility of performing network management operations through web applications. Concretely, RESTCONF provides a way to realize CRUD (Create, Retrieve, Update, Delete) operations through execution of HTTP methods to access to configuration data defined in YANG, using the datastore concepts defined in NETCONF.

- ***CPE WAN Management Protocol (CWMP)***
  CWMP is a protocol defined by Broadband Forum in TR-069 Technical report in order to remotly manage customer-premises equipment (CPE) connected to an Internet Protocol (IP) network. This protocol allows performing tasks such as auto-configuration, software or firmware image management, software module management, status and performance management, and diagnostics.

- ***OMA-DM***
  OMA-DM is a secure device management protocol specified by the Open Mobile Alliance (OMA) Device Management (DM) Working Group and the Data Synchronization (DS) Working Group. It enables performing management tasks such as device provisionning, device configuration, software upgrade and fault management.

Nevertheless, the above network management protocols were designed before the emergence of IoT paradigm, and it was rather obvious that those protocols did not consider a number of IoT characteristics and constraints (e.g. devices resource constraints) that raise technical barriers for their applicability in the IoT environment. Therefore, new network management protocols and frameworks have been proposed in order to cope with the inherent constraints of IoT environment. We discuss about those solutions in the next section.

2.3 Management of IoT low power networks

In order to fulfill the requirement of IoT low power networks mentioned in section 1.5, several IoT networks management solutions have been proposed for resource-constrained devices. These network management solutions includes new network management protocols, SDN based frameworks, Cloud based frameworks, Semantic based frameworks and machine learning based frameworks.

2.3.1 Network management protocols

In the literature, different network management protocols have been proposed in order to remotely manage resource-constrained devices. These protocols include: LWM2M, CoMI, NETCONF light and 6LowPAN-SNMP.

- ***LWM2M***
  LWM2M is a client-server protocol developed for the management of IoT low power devices. This protocol has been designed by Open Mobile Alliance (OMA) and is based on protocol and security standards from the IETF. It provides several features such as connectivity monitoring, resources monitoring and firmware upgrade. In Figure 2.2, we depict a high-level view of LWM2M architecture. LWM2M server is located at the network manager device and LWM2M client are typically located on managed devices. IoT device resources are organized into objects (e.g. Location object contains all resources that provide information about the location of IoT devices). A description of the implementation of that protocol is provided by the authors in [125].

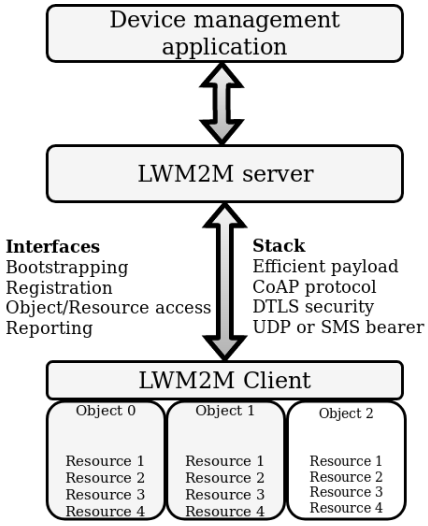


Figure 2.2 – LWM2M architecture.

- ***CoAP Management Interface(CoMI)***
  CoMI is a management interface dedicated for IoT low power devices and networks. This network management protocol enables performing management operations on IoT device resources specified in YANG, or SMIv2 converted to YANG by accessing them through the CoAP protocol. The specification of that protocol is given in [155].

- ***6LowPAN-SNMP***
  6LowPAN-SNMP is an adaptation of SNMP for IPv6 Low-Power Wireless Personal Area Network (6LowPAN) proposed in [33]. It has been designed to work in resource-constrained networks and offers the possibility to perform SNMP operations over IPv6 Low-Power Wireless Personal Area Networks. To achieve that, a mechanism of compression of SNMP header is performed in order to reduce the number of SNMP messages generated. The compatibility with standard SNMP is ensured by using a proxy forwarder that helps to convert SNMP messages into 6LowPAN-SNMP messages.

- ***NETCONF light***
  NETCONF light is a network management protocol developed by IETF [133] in order extends NETCONF to enable the management of resource-constrained devices. It provides tools to install, manipulate, and delete the configuration of network devices by using only a set of NETCONF operations.

It is worth mentionning that network management protocols are often associated with messaging protocols in order to enable the management of resource-constrained devices [97, 131]. These messaging protocols include: CoAP (Constrained Application Protocol) [139], XMPP (Extensible Messaging and Presence Protocol) [129], DDS (Data-Distribution Service for Real-Time Systems) [63], MQTT (Message Queuing Telemetry Transport) [109], MQTT-SN (MQTT for Sensor Networks) [146] and AMQP (Advanced Message Queuing Protocol) [23]. We provide a comparison of those messaging protocols in Table 2.1.

In Table 2.2, we summarized a comparison of different protocols for management of resource-constrained networks according to the requirement formulated in section 1.5. Nevertheless, these network management protocols are not able to satisfy all the requirements of IoT low power networks mentioned earlier. To achieve that, it is necessary to associate those protocols with other mechanisms to fulfill the requirement such as self configuration and scalability. In the next section we will discuss about those mechanisms.

Protocol	Suitability for Constrained devices	Messaging type	Architecture	QoS	QoS Level	Interoperability
CoAP	+++	- Publish/ subscribe - Request/ Response	Client/Server	Yes	- Confirmable message - Non-confirmable message	Yes
AMQP	+	- Publish/ subscribe	Client/Server	Yes	- At-most-once - At-least-once - Exactly once	Yes
MQTT	++	- Request/ Response - Publish/ subscribe	Client/Broker	Yes	- QoS 0 (fire and forget) - QoS 1 (delivered at least once) - QoS 2 (delivery exactly once)	Partial
MQTT-SN	++	- Request/ Response - Publish/ subscribe	Client/Broker	Yes	- QoS 0 (fire and forget) - QoS 1 (delivered at least once) - QoS 2 (delivery exactly once)	-
XMPP	+	- Request/ Response - Publish/ subscribe	Client/server	No	-	Yes
DDS	+	Publish/ subscribe	Brokerless	Yes	23 levels of QoS	Yes
+++ Excellent, ++ Fair, + Poor						

Table 2.1 – Messaging protocols used in IoT networks.

Network management protocol	Scalability	Fault tolerance	Energy efficient	QoS	Security	Self configuration
LWM2M	-	-	√	-	√	-
CoMI	-	-	√	-	√	-
6LowPAN-SNMP	-	-	√	-	√	-
NETCONF light	-	-	√	-	√	-
√ The requirement can be handle by the Network management protocol.						

Table 2.2 – A comparison of IoT low power networks management protocols.

2.3.2 IoT network management based on Cloud frameworks

According to [105], cloud computing refers to a model which enables ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort, often over the internet. These services are provided through a cloud platform. A cloud platform according to a definition given in [11], is "a unique sensor data storage, visualization and remote management platform that leverage [sic] powerful cloud computing technologies to provide excellent data scalability, rapid visualization, and user programmable analysis.". Cloud computing infrastructure can provide needed resources for data storage or computing power to IoT low power networks. For this reason, various project aiming implementation of sensor cloud have been conducted by researchers [67]. Additionally, several clouds of things platform have emerged to foster blend of cloud paradigm and IoT networks in order to enable management of connected devices over the cloud platform.

Table 2.3 presents some cloud IoT platforms from the literature and their characteristics. Generally, the architecture for management of IoT low power networks over a cloud plateform consists in three levels: 1) the first level is composed of resource-constrained devices, 2) the second level is composed by cloud infrastructure and 3) the third level is composed of IoT applications. We provide an example of such architecture in Figure 2.3.

Yuriyama and Kushida [175] proposed a management solution for sensors network based on a cloud infrastructure. In the proposed solution, physical sensor devices are virtualized in order to enable the management of heterogeneous resource-constrained devices over a cloud platform infrastructure. Likewise, Xu and Helal [167] proposed an architecture for management of IoT devices called Cloud-Edge-Beneath (CEB). This proposal leverage the benefits of cloud platforms in order to

IoT cloud platform	Protocols for data collection	Configuration management	Device Monitoring	Communication technologies	Resource constrained devices
Mbed [16]	HTTP, HTTPS, CoAP, MQTT	√	-	BLE, Thread, 6LowPAN, Wi-Fi, LoRa	√
Arkessa [15]	-	√	-	-	-
Xively [100]	MQTT, CoAP, HTTP	√	√	Thread, LoRa, sigfox	√
Thethings.io [150]	HTTP, CoAP, MQTT, WebSockets	√	√	Sigfox, Wi-Fi, LoRa, GSM	√
Arrayent [148]	-	√	-	-	-
ThingWorx [152]	-	√	√	-	-
Autodesk Fusion Connect [19]	CoAP, HTTP, XMPP, DDS, MQTT	√	√	-	-
IBM IoT [72]	MQTT, HTTP	√	-	-	-
Artik Cloud [130]	REST/HTTP, websockets, MQTT, CoAP	√	-	-	-
Carriots [31]	HTTP, MQTT	√	-	-	-
Echelon [46]	-	√	-	WiFi, Ethernet	-
KAA [81]	MQTT, CoAP, XMPP, TCP, HTTP	√	√	Z-Wave, Zigbee, LoRa, Bluetooth, WiFi, 6LoWPAN	√
Ayla IoT Platform [111]	-	√	-	Wi-Fi, Ethernet, Zigbee	√
Thinger.io [151]	MQTT, CoAP and HTTP	√		Sigfox	√
SiteWhere [143]	MQTT, AMQP, Stomp, WebSockets, and direct socket connection	√	-	-	-

Table 2.3 – Cloud of Things platforms features.

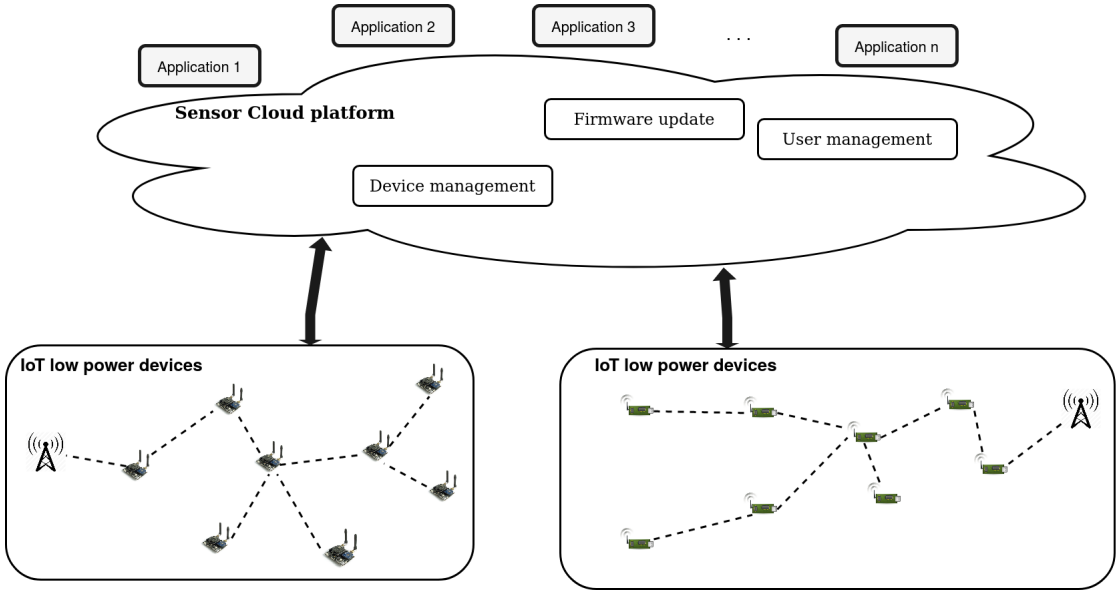


Figure 2.3 – Example of architecture for management of IoT devices over a sensor cloud infrastructure.

provide a management solution for large-scale and dynamic IoT networks. Similarly, Ojha et al. [112] proposed a solution for management of wireless sensor networks based on a cloud platform. The proposed solution enables dynamic scheduling of duty in order to extend the network lifetime. Along the same lines, Kim et al. [88] proposed a routing scheme called H-SMSR (hierarchical scalable multipath source routing) in the context of IoT low power devices managed over a cloud platform called Agriculture Sensor-Cloud Infrastructure (ASCI). The proposed routing protocol includes hierarchical source routing (HSR) and aggregation gradient routing (AGR) in order to increase the network lifetime. Das et al. [38] proposed an energy efficient model for the management of IoT low power devices over a cloud platform. This solution includes a predictive model that helps to reduce the network transmission overhead. Suciu et al. [147] proposed a framework based on a cloud platform to enable management of IoT low power devices in the context smart cities. The proposed framework allows an improvement of the network traffic quality through an autonomic management of IoT devices.

The above frameworks for management of IoT low power networks exhibit different functionalities. We summarized those frameworks in Table 2.4 and evaluated them against the requirements of IoT low power networks formulated in section 1.5. From the table, we see that none of the existing solutions fulfill all the requirements of IoT low power networks. Therefore, additional mechanisms are needed in order to meet requirements of IoT low power networks.

Network management framework	Scalability	Fault tolerance	Energy efficient	QoS	Security	Self configuration
[175]	-	-	√	-	-	√
[167]	√	-	√	-	-	√
[112]	-	-	-	√	-	√
[88]	√	-	√	-	-	√
[38]	-	-	√	-	-	-
[147]	√	-	√	-	√	√
√ The requirement can be handle by the network management framework.						

Table 2.4 – A comparison of IoT network management frameworks based on Cloud.

2.3.3 IoT network management based on SDN frameworks

Over the last decade, the number of resource-constrained devices present in the IoT ecosystem has increased dramatically. These devices are often running many events which imper on the network performance. To cope with this issue, Software Defined Network (SDN) has been used in order to achieve energy efficient management of IoT low power networks [110]. According to [87], SDN is a paradigm where a central software program called a controller dictates the overall network behavior. SDN advocates separating control plane of the network (where decisions about how packets should flow through the network is taken) from the data plane of the network (traffic forwarding plan). Figure 2.4 gives an overview of an SDN architecture. Network devices are considered as simple packet routing devices (data plan) and the control logic is implemented at the controller (control plane). Southbound interface is the relay between programmable switches and the software controller. Several southbound API has been proposed in literature [135], notably: OpenFlow, ForCES, PCEP, etc. Openflow [94] is considered as the most common southbound SDN interfaces. Openflow exists in several software releases [137] : NOX, POX, Beacon, Floodlight, MuL, Maestro, Ryu, etc. Northbound interface enables communication among the higher-level components. In fact, northbound interface allows exchange of information between network and application running on top of it.

De Gante et al. [40] proposed a centralized architecture for the management of wireless sensor networks. The proposed architecture leverages the benefits of SDN paradigm, notably it allows prolonging the network lifetime. In the same vein, Costanzo et al. [37] and Jacobsson and Orfanidis [77] proposed network management solutions for resource-constrained devices based on SDN. Moreover, Orfanidis [114] proposed an architecture for management of IoT low power networks

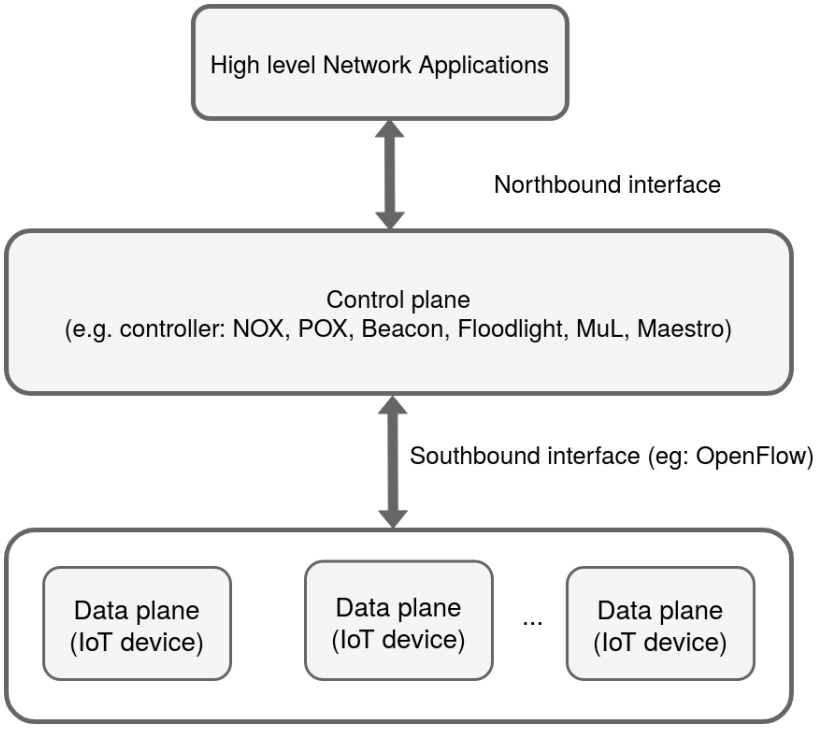


Figure 2.4 – SDN architecture.

based on SDN with a machine learning model. Similarly, Bera et al. [21] proposed a centralized network management scheme called software-defined wireless sensor network architecture (Soft-WSN) in order to configure IoT low power networks according to policies defined by the network management entity. Huang et al. [69] proposed a framework for management of IoT low power networks based on SDN and reinforcement learning. The proposed framework enables reduction of the overhead of control traffic by filtering redundancy and performing a load-balancing routing mechanism according to data flows with the required QoS. Additionally, Wu et al. [166] proposed a framework based on SDN to mitigate security attack in wireless sensor networks. The proposed framework enable dynamic reaction against unknown attacks. However, since these solutions are centralized, in a large network, they may suffer from scalability problem. To cope with this issue, Olivier et al. [113] proposed an architecture called software-defined clustered sensor networks (SDCSN). The proposed network management framework uses clustering technique to organize the network in clusters where each cluster head plays the role of the controller. In the same line, Oliveira et al. [41] proposed an implementation of a scalable framework for management of wireless sensor networks based multiple SDN controller.

We provide a comparison of the above frameworks in Table 2.5 according to the requirement formulated in section 1.5. It is worth mentioning that SDN needs to be associated with another mechanisms [102] such as machine learning in order to

fulfill the requirements of IoT low power networks.

Reference	Scalability	Fault tolerance	Energy efficient	QoS	Security	Self configuration
[40]	-	-	√	-	-	-
[37]	-	-	√	-	-	-
[77]	-	-	√	-	-	-
[113]	√	√	√	√	√	-
[41]	√	-	√	-	-	-
[166]	-	-	√	-	√	√
[21]	-	√	√	-	-	√
[114]	-	√	-	-	-	√
[69]	-	√	√	√	-	√
√ The requirement can be handle by the network management framework.						

Table 2.5 – A comparison of IoT network management frameworks based on SDN.

2.3.4 IoT network management based on Semantic technologies

The presence of billion of heterogeneous and resource-constrained devices in IoT environment raises the need for handling heterogeneity of devices management solutions. For this purpose, semantic technology has been used to cope with IoT devices heterogeneity while ensuring good network performance.

Katasonov et al. [83] present a middleware for self management of heterogeneous IoT devices. This middleware is based on agent technologies and it enables interoperability by using semantic technologies. Vlacheas et al. [156] proposed a framework for management of IoT devices deployed in context of smart cities applications. The proposal is based on the concept of cognition and proximity and provides mechanisms to face heterogeneity of connected things. In the same vein, Ismail et al. [76] proposed a framework based on semantic technology in order to enable management of IoT devices. The proposed framework ease automatic management of IoT devices by using ontology to enable management of heterogeneous network devices. Likewise, Sahlmann et al. [128] proposed a framework based on the oneM2M ontology (a structured vocabulary that describes a certain domain of interest) in order to ease automatic configuration of heterogeneous IoT devices. The proposed solution uses NETCONF and MQTT protocols for management of resource-constrained devices. Further, Datta et al. [39] proposed a framework based on semantic technologies to enable management of heterogeneous

IoT devices. The proposed framework includes automatic discovery of the mobile devices, provisioning of sensors and IoT domains, semantic reasoning on sensor data and actuation based on the suggestions. The authors claim that their proposal can help to efficiently use the resources of IoT devices. In the same vein, Aïssaoui et al. [6] proposed an extension of SAREF ontology in order to manage heterogeneous IoT devices. The proposed model enables cross-system data interoperability and knowledge enrichment through reasoning.

Based on this state of the art on IoT networks management based on semantic, we observed that existing solutions focused on enabling automatic management of heterogenous IoT devices. However, in order to meet the requirement of IoT low power networks formulated in section 1.5, those solutions should be enhanced. We summarize in Table 2.6 a comparison of these solutions.

Network management framework	Scalability	Fault tolerance	Energy efficient	QoS	Security	Self configuration
[156]	-	-	-	-	-	√
[76]	-	-	-	-	-	√
[39]	-	-	√	-	-	√
[128]	-	-	-	-	-	√
[6]	-	-	-	-	-	√
[83]	-	-	-	-	-	√
√ The requirement can be handle by the network management framework.						

Table 2.6 – A comparison of IoT network management frameworks based on Semantic.

2.3.5 IoT network management based on Machine Learning techniques

Nowadays, IoT networks generate a huge amount of data due to the dynamic nature of these networks and/or the number of resource-constrained devices. In order to leverage the benefits of those data, machine learning techniques have been used in order to help in taking decision of network management [14, 160, 93]. Machine Learning (ML) refers to the process that gives a computer the ability to mimic the human brain in order to perform complex tasks based on their knowledge. It has been useful for IoT network management because it provides predictive mechanisms that help taking decision such as routing table reconfiguration, network scheduling, parameter adaptation according to the current states of the network. In general, ML techniques can be divided into supervised learning, unsupervised learning and

reinforcement learning as depicted in Figure 2.5. Supervised Learning is a ML method which provides a way to predict the outcome of unseen values by using classification or regression with pre-labelled data. It is based on two steps namely training (phase which involves dataset training and designing of classification model) and testing (which involves classification of unseen value). The common supervised learning algorithms used for IoT network management include : Support Vector Machine (SVM), regression tree, neural network, Convolutional neural network (CNN), Deep Belief Network (DBN) and Recurrent Neural Network (RNN). Unlike supervised learning, unsupervised learning is not based on pre-labeled. It uses instead unlabeled dataset to perform classification of data into cluster by discovering common pattern within those unlabeled dataset. The common unsupervised learning algorithms used for network management include: K-MEANS, Autoclass, Deep Belief Network and Deep Boltzmann machine. Reinforcement learning is another approach of ML that enables to find the ideal behavior in particular context by machines and software agent in order to maximize performance. Basically, the reinforcement learning is described as a Markov Decision Process (MDP). Figure 2.6 shows a high level overview of a reinforcement learning model. The agent can visit a set of finite environment states S by performing actions. In visiting a state, a numerical reward will be collected in order to measure the success or failure of an agent's actions in a given state. The common reinforcement algorithms used for IoT network management include: Sarsa, Q-learning and Policy Gradient.

Generally, the usage of ML for solving networking problem is done by following specific steps as shown in [160]. These different steps are described in Figure 2.7 and include problem formulation, data collection, data analysis, model construction, model validation, deployment and inference.

In the following, we present some existing solutions for management of IoT low networks based on ML.

- ***IoT network management solutions based on supervised learning***
  Wang et al. [162] proposed a framework based on decision tree learners, a supervised learning algorithm, in order to predict the link quality in IoT low power networks. The proposed solution aims to optimize the network performance by taking routing decision that helps improving the data delivery rate and data latency. Likewise, Liu and Cerpa [98] proposed a framework called 4C, to estimate the link quality in IoT low power networks. The proposed framework is based on logistic regression and uses PHY parameters of the last received packets and packet reception rate (PRR) to estimate the link quality. The authors claim that very little data (5-7 links for a couple of minutes) are needed in order to train the models in the environments tested.

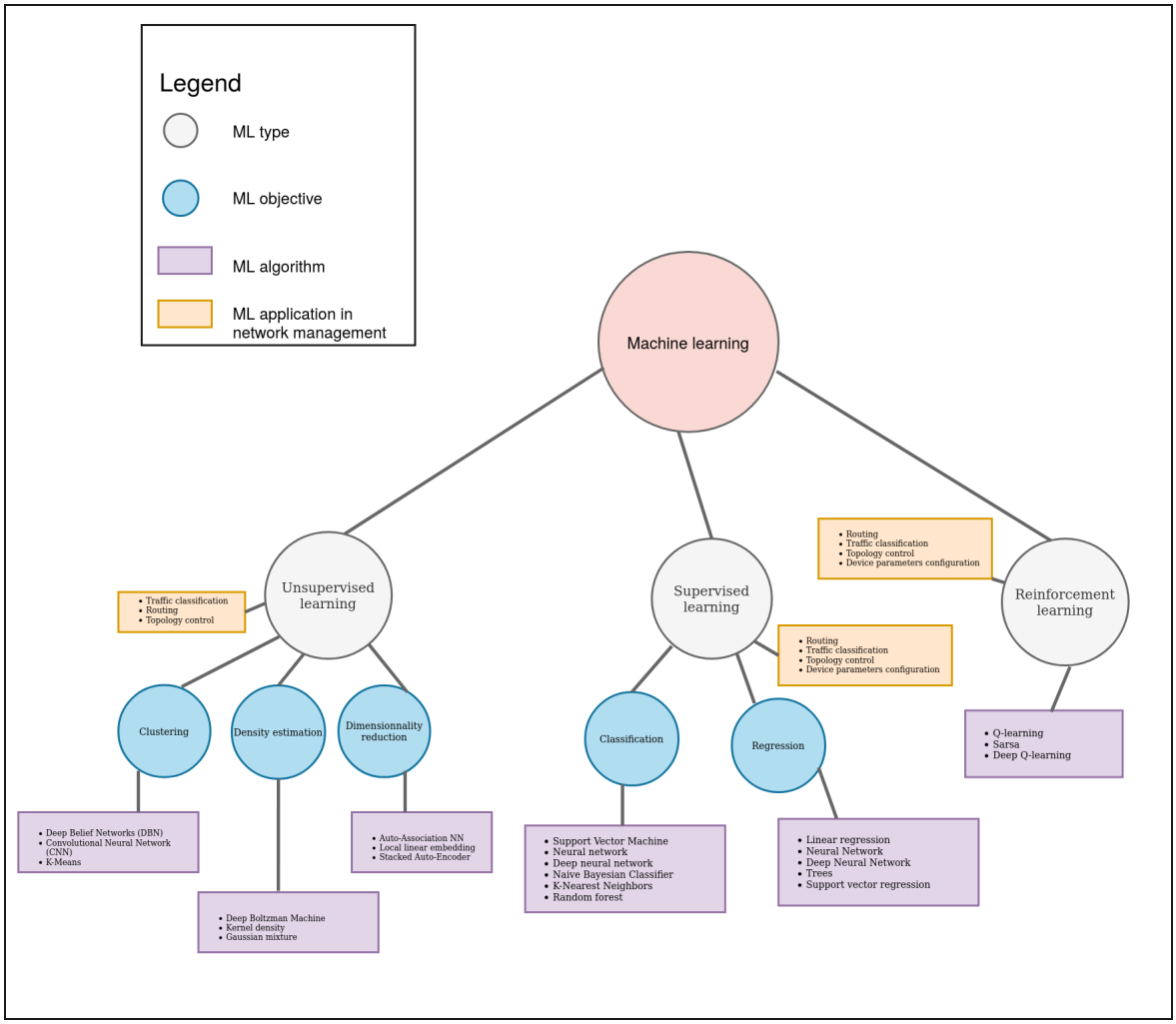


Figure 2.5 – Different Machine learning algorithms used for IoT network management.

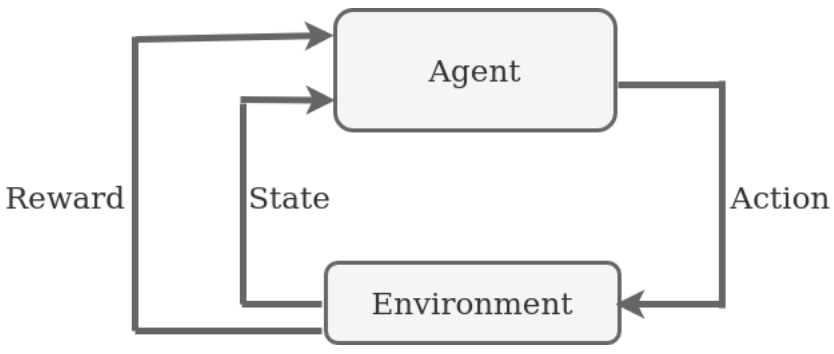


Figure 2.6 – Reinforcement learning model.

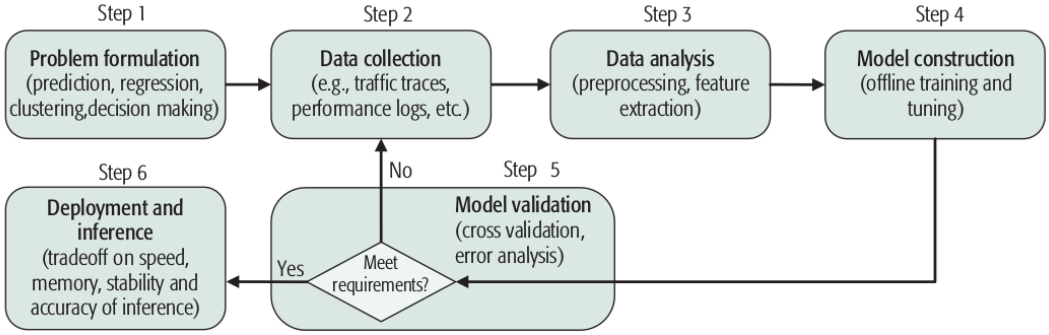


Figure 2.7 – Workflow of machine learning for networking [160].

In the same vein, Feo-Flushing et al. [51] presented a scheme to perform an online learning using a supervised learning algorithm in order to predict the link quality in a given wireless sensor network. The authors claim that strategies that keep balanced the set of training samples in terms of ranges of target values provide better accuracy and faster convergence. Further, Kaplantzis et al. [82] proposed a centralized intrusion detection system (IDS) based on Support Vector Machines(SVMs) and sliding windows for wireless sensor networks. The proposed IDS uses only 2 features to detect selective forwarding and black hole attacks.

- ***IoT network management solutions based on unsupervised learning*** Barbancho et al. [20] proposed a solution called Intelligent Wireless Sensor Network (IWSN) in order to manage data route by IoT devices. The proposed solution is based on neural network which allows the selection of the route that optimize the network performance in presence of node failure. Additionally, Moustapha and Selmic [108] proposed a dynamic model for fault detection in wireless sensor networks. The proposed framework includes neural network modeling approach for sensor node identification and fault detection in the network. Further, Branch et al. [28] proposed a method for outlier detection method in WSNs using k-nearest neighbors. The authors claim that their proposal is well suited for applications in which the confidence of an outlier rating may be calculated by either an adjustment of sliding window size or by the number of neighbors used in a distance-based outlier detection technique. Another framework for management of IoT low power networks was proposed by Chang et al. [32]. The proposed framework aims at controlling the topology of ultra-dense wireless sensor networks. The proposed framework is based on K-Means, an unsupervised learning algorithm, and it enables an optimization of the network lifetime by balancing energy consumption.

- ***IoT network management solutions based on reinforcement learning*** Stampa et al. [145] proposed a framework based on Deep-Reinforcement Learning (DRL) agent for routing optimization. The proposed framework helps to define tailored configurations that minimize the network delay. Additionally, Shah and Kumar [136] proposed a framework called Distributed Independent Reinforcement Learning (DIRL), for resource/task management in wireless sensor networks. The proposed framework is based on Q-learning and it allows each sensor device to autonomously schedule its tasks and allocate its resources by a learning process. Another framework for management of IoT low power networks based on reinforcement learning was proposed by Mihaylov et al. [106]. The proposed framework enables scheduling the wake-up cycles of nodes in a wireless sensor network according to their interactions with neighbouring nodes. Further, a framework based on reinforcement learning for routing management in wireless sensor has been proposed in [161]. The proposed framework called AdaR (Adaptive Routing) uses Least Squares Policy Iteration (LSPI) and allows sensor nodes to learn the optimal routing strategy regarding a set of optimization goal. Furthermore, Förster and Murphy [54] proposed a framework called Feedback ROuting to Multiple Sinks (FROMS), to optimize routing selection in wireless sensor network. The proposed framework based on reinforcement learning helps to define the optimal multicast routes using different cost metrics (e.g. hops, geographic distance, latency and remaining battery). Moreover, FROMS enables quick recovery in case of failures and sink mobility.

We summarized frameworks for IoT low power networks management based on machine learning in Table 2.7 and compared them according to the requirement of IoT low power formulated in section 1.5. From this study, we can see that additional effort are needed to develop efficient solutions in order to meet the requirement of IoT low power networks.

Network management framework	Scalability	Fault tolerance	Energy efficient	QoS	Security	Self configuration
Supervised Learning						
[162]	-	√	-	√	-	√
[82]	-	-	-	-	√	√
[98]	-	√	-	-	-	√
[51]	-	√	-	-	-	√
Unsupervised Learning						
[20]	-	√	√	√	-	√
[108]	-	√	-	-	-	√
[28]	-	-	-	-	√	√
[32]	-	-	√	-	-	√
Reinforcement Learning						
[145]	-	-	-	√	-	√
[136]	-	-	-	√	-	√
[106]	-	-	√	-	-	√
[161]	-	-	√	-	-	√
[54]	√	√	√	-	-	√
√ The requirement can be handle by the network management framework.						

Table 2.7 – A comparison of IoT network management frameworks based on Machine learning.

2.4 Conclusion

In this chapter, we presented a comprehensive overview on existing solutions for management of IoT low power networks and provided a comparison of those solutions according to the requirement of IoT low power networks. From this review, we can see that there is still research effort to be done in order to meet the requirement of IoT low power networks. In the remainder of this thesis, we will present improvement of these solutions with our contributions presented in chapter 3, chapter 4, chapter 5 and chapter 6.

Part II

Contributions

Efficient transmission range for IoT low power networks

Sommaire

3.1 Introduction

Nowadays, IoT low power networks have been largely adopted in various scenarios such as the factory of the future, smart agriculture, smart cities, and so forth. These networks allow the interconnection of resource-constrained devices (e.g. actuators and sensors) using wireless communication links. In such networks, having energy-efficient routing protocols is necessary in order to increase the battery lifetime of devices. However, many factors may influence the energy consumption in IoT low power networks. These factors include the transmission power of devices, the data routing strategy, etc. One of the most widely adopted routing protocol for those networks is the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL). RPL is a proactive routing protocol that enables the construction and maintenance of dynamic routes using various objective functions (OFs) [165]. This protocol constructs the network topology as a Destination Oriented Directed Acyclic Graph (DODAG). Each node within the network selects a preferred parent among a list of available parents to route data toward the RPL root. Nevertheless,

RPL performance can be negatively affected by frequent changes of the preferred parents [73, 3]. Although those changes are driven by the OF function, they can substantially reduce the performance of the RPL protocol by inducing high communication overheads and excessive energy consumption because of the required path reconstruction. In the literature, various improvements of RPL objective functions have been proposed in order to improve the network performance. Aslani and Sargolzaey [18] proposed a new approach to compute the RPL routing metric ETX (Expected Transmission Count) in order to optimize the routing path selection so as to improve the RPL network performance. In the same vein, Yang et al. [169] proposed a routing metric named Stability Index (SI) to improve RPL routing table stability. The proposed metric uses control message transmission rate to measure the stability of node and routing topology. Based on that metric, more stable nodes are selected to form more stable routes. Likewise, Iova et al. [74] proposed a routing metric called Expected Lifetime metric (ELT) in order to improve the network lifetime of RPL networks. The proposed routing metric helps to create an energy balanced topology. Furthermore, Gaddour et al. [55] proposed a new objective function called fuzzy logic objective function (OF-FL). The proposed OF is based on fuzzy logic and it uses several important routing metrics to enable route selection in RPL networks.

However, in the aforementioned approaches, the efficiency of the best parent selection-based methodology to ensure energy-efficient routing in RPL networks is questionable since these approaches do not consider the impact of the network transmission range on their intrinsic performance. In other words, existing approaches may fail to minimize the network energy consumption whenever the transmission range value is not well chosen, resulting in a non-efficient energy consumption in the RPL network. In practice, varying the transmission range is done by varying the transmission power.

Nevertheless, the evaluation of the efficient transmission range for data routing, given an RPL network topology, is challenging since short or long transmission range values cannot guarantee a good network performance. Indeed, as pointed out by [104], simply setting the transmission range, for data routing, to a minimal or maximal value does not guarantee network lifetime maximization. In fact, having a short transmission range may generate excessive energy consumption by intermediate hops in a multi-hop communication network. Likewise, having a long transmission range may require an important energy consumption for data routing. Therefore, given an RPL network topology, an efficient transmission range needs to be calculated so that the network lifetime will be maximized.

Kubisch et al. [92] proposed a method that helps to dynamically adjust the

transmission power level (or transmission range) of IoT low power networks in a distributed fashion. Additionally, Zhang and Gorce [177] proposed an energy efficiency metric for periodic monitoring applications called the Energy Distance Ratio per bit (EDRb). The proposed metric enables to determine the transmission range that minimizes the energy consumption in wireless sensor networks (WSNs). Similarly, Deng et al. [44] proposed a per hop method to determine the transmission range that miminizes the network energy consumption in IoT low power networks. Further, Yi et al. [173] proposed a method to control the transmission range for energy-harvesting wireless sensor networks. The proposed method helps to increase the transmission range when a surplus of energy is generated to decrease the number of relay hops. In these approaches, data routing is based on the assumption that each node is only aware of the nodes within its transmission range. With such distributed routing path construction strategy, the resulting network routing infrastructure may comprise disjoint network segments, making end-to-end data routing impossible.

Song et al. [144] show that the evaluation of the efficient transmission range of sensor nodes is NP Hard. For such type of problem, machine learning techniques may be a good candidate to solve it [2].

In this chapter, we address the effect of the transmission power on energy consumption in RPL networks (RPL : IPV6 Routing Protocol for Low-Power and Lossy Networks) and propose new solutions based on a multi-layer perceptron (MLP) model to estimate the transmission range of RPL networks that reduces the network energy consumption while preserving the network connectivity. In particular, we propose intelligent solutions to determine the efficient transmission range for:

- RPL networks with a fixed number of nodes;
- Dynamic RPL networks (where the network size may vary with the time because of node failure, adding or removing nodes to/from the network and so forth) deployed following a 2D topology;
- Dynamic RPL networks deployed following a 3D topology.

The reminder of this chapter is organized as follows. Section 3.2 presents background on RPL and MLP; section 3.3 presents our solution to determine the efficient transmission range of RPL networks with a fixed size; section 3.4 presents our solution to determine the efficient transmission range of dynamic RPL networks; section 3.5 presents a discussion concerning the deployability of our solution; section 3.6 gives concluding remarks.

3.2 Background

3.2.1 RPL networks

RPL constructs the network topology (or DODAG) based on a rank of a node (the position of a node in relation to the RPL root) which is computed using an objective funtion. The objective function allows to define how an RPL node selects a preferred parent for data routing by using one or more routing metrics (e.g. Hop Count, Link Throughput, Link Quality Level, node remaining energy, etc.). The establishment of a DODAG is done using these control messages:

- DIS (DODAG Information Solicitation): this message is used to request information (e.g. DODAG Information Object) from neighboring RPL nodes. It allows a node to discover new routes.
- DIO (DODAG Information Object): this message is used to define upward routes. This message contains informations (DODAG identifier, the Objective Function, the Rank of the node and so forth) that allow a node to discover an RPL Instance.
- DAO (Destination Advertisement Object): this message is used to define downward routes.

Figure 3.1 shows an example of RPL DODAG construction. After the convergence of RPL, the protocol will continue to produce control messages in order to maintain the routing tables up-to-date. This may generate an excessive energy consumption whenever the transmission range of RPL nodes is not well chosen [68, 3].

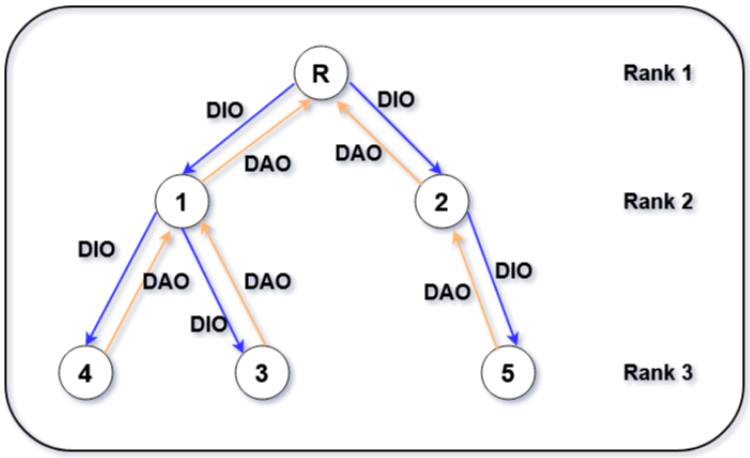


Figure 3.1 – Example of DODAG construction.

3.2.2 Multilayer perceptron

The multilayer Perceptron (MLP) is a deep neural network model which allows to perform classification and regression on a given set of historical data. It has been used in IoT low power networks to solve tasks such as localization and intrusion detection. Typically, the MLP model is composed of an input layer, an output layer and at least one hidden layer as depicted in Figure 3.2. The input layer is an M-dimensional vector that takes the values of the input features, denoted by $V = (x_1, y_1, x_2, y_2, \cdots, x_M, y_M)$. In our case, the vector V contains the coordinates of each node for a given RPL network topology. The inputs are propagated through the hidden layer(s) towards the output layer. The hidden layer is located between the input layer and the output layer. The output h_i^j of the neurons in the hidden layer is defined as follows [124]:

$$h_i^j = f\left(\sum_{k=1}^{n_{i-1}} w_{k,j}^{i-1} h_{i-1}^k\right) \tag{3.1}$$

For $i = 2, \cdots, N$ and $j = 1, \cdots, n_i$.

Where $w_{k,j}^{i-1}$ is the weight between the neuron k in the hidden layer i and the neuron j in the hidden layer $i+1$. n_i is the number of neurons in the ith hidden layer.

The output layer is a vector obtained through the following equation:

$$y_i = f\left(\sum_{k=1}^{n_N} w_{k,j}^{N} h_N^k\right) \tag{3.2}$$

Where $w_{k,j}^{N}$ is the weight between the neuron k in the Nth hidden layer and the neuron j in the output layer. n_N is the number of the neurons in the Nth hidden layer.

When the MLP is applied to a supervised learning problem, it generally performs training using the backpropagation algorithm [78] in order to generate a model that can accurately realize the task for which it is being trained. In our proposal, the goal is to accurately estimate the efficient transmission range for a given RPL network topology based on a set of data collected beforehand (data composed of network topologies and their associated efficient transmission range).

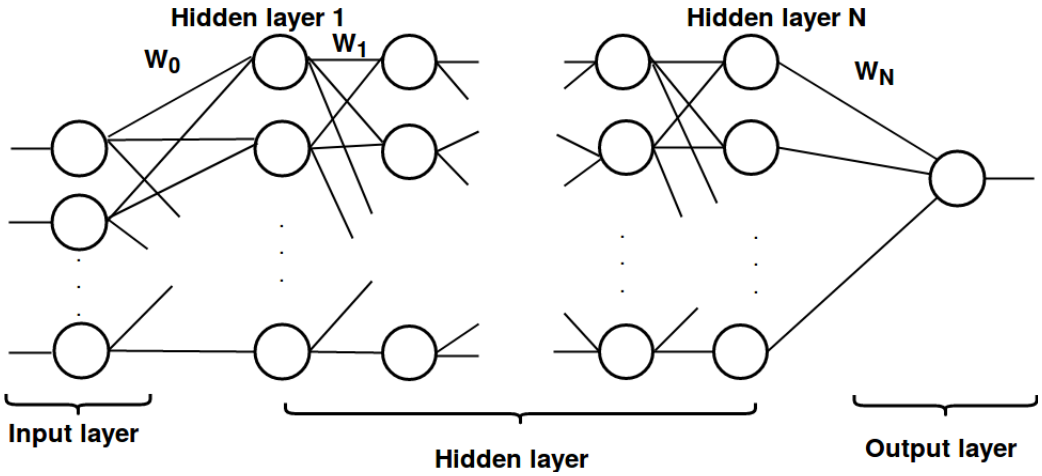


Figure 3.2 – Example of an architecture of a Multilayer Perceptron.

3.3 Efficient transmission range for static RPL networks

3.3.1 Assumptions and problem description

3.3.1.1 Assumptions

We considered an RPL network composed of k nodes randomly deployed in an area of interest. The set of nodes is represented by $D = \{d_1, d_2, \cdots, d_k\}$. Each node has a unique identifier i in the network ($i \in \{1, \cdots, k\}$). Also, each RPL node can communicate with another node if the node is within its transmission communication range. Moreover, we considered that all the devices are homogeneous in terms of wireless communication technology, computational capabilities and have the same amount of initial energy. Furthermore, we assumed that the nodes are topologically static.

3.3.1.2 Problem description

We conducted a two-step investigation on a Cooja emulator [115] running an RPL network composed of 50 motes. In the first step, we used the default communication range of 50 meters and observed the control messages generated. In the second step, we manually reduced the communication range to 20 meters and observed also the control messages generated with this new parameter value. After one hour, we noted that the number of control messages in the second step is significantly lower than the one of the first step (cf. Figure 3.3). Thus, in the second step, few control messages (which implies less energy consumption) are needed to discover and maintain the routes. However, as shown in [104], a minimal value of transmission range may not

ensure an optimal network performance. Thereby, our goal here is to determine the efficient transmission range that helps to enhance the performance for static RPL networks with a defined number of nodes.

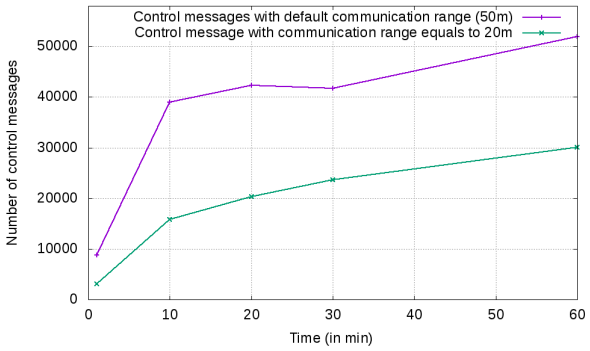


Figure 3.3 – Comparison of control messages transmitted.

3.3.2 Description of our solution

The main idea behind our solution that enables an adjustment of the transmission range of an IoT network which implement RPL is to ensure the reduction of the network energy consumption and a maintain of the network connectivity. To achieve that, our solution uses nodes position in order to estimate the efficient transmission range through a neural network.

Our proposed solution to estimate the transmission range of RPL networks with a fixed number of nodes works in three steps. The first step is the data collection phase, which is done on Cooja emulator. The second step consists in building, offline, the learning model. Finally, the third step consists in testing our model on a set of new network topologies.

In order to generate the dataset to construct a predictive model that determines the efficient transmission range of static RPL networks, we setup a network of 30 nodes, running RPL protocol using Contiki Cooja tool [115]. These nodes are periodically sending messages to the sink node, which means that a certain amount of energy is consumed in that RPL network. For this network setting, we measured the total energy consumed by all the RPL nodes. The simulation parameters are depicted in Table 3.1.

Algorithm 3.1 shows the pseudo code of the labeling process. The inputs of our dataset are composed of the positions of the different nodes, and the output is the efficient transmission range. This transmission range is chosen such that each node is reachable by the DODAG root (either directly or through multiple hops) while the total energy consumption of the RPL network is minimized.

Table 3.1 – Simulation parameters.

Parameter	**value**
Deployment area (m^2)	100×100
Number of nodes	30
Initial Energy(μAh)	1000000
Packets rate	1 packet/60s
Simulation time (ms)	1200000
Successful transmission ratio (Tx)	0.9
Successful receiving ratio (Rx)	1.0
Transmission range (m) (Tx)	20,30,40,50,60,70,80,90,100

We performed the training of our neural network model based on the collected dataset using WEKA [64]; a well-known machine learning library that contains tools for data preparation, classification, regression, etc. Figure 3.4 shows the structure of our MLP, which consists of an input layer, two hidden layers and a single output layer.

Algorithm 3.1: Output labeling

$n \leftarrow 100$
$T = [t_1, t_2, ..., t_n]$ // a set of topologies
for t_i in T **do**
  $bestTransmissionRange \leftarrow P_i$
  $DefaultEnergyConsumption \leftarrow \theta_i$
  for P_j in $\{20, 30, 40, 60, 70, 80, 90, 100\}$ **do**
    if (all motes have parent) **then**
      if ($E_{consumed} < DefaultEnergyConsumption$) **then**
        $bestTransmissionRange \leftarrow P_j$
        $DefaultEnergyConsumption \leftarrow E_{consumed}$
      end if
    end if
  end for
  dataset.add($t_i, bestTransmissionRange$)
end for
return *dataset*

3.3.3 Performance evaluation

To illustrate the performance of our trained model, we have deployed 30 Cooja motes on an area of 100 x 100 m^2 and generated a set of 5 topologies for testing: T0,..., T4. We used nodes position to get the efficient transmission range through our machine learning model. Then, we performed simulation using Cooja network emulator on Contiki 3.0. For the performance evaluation, we have considered

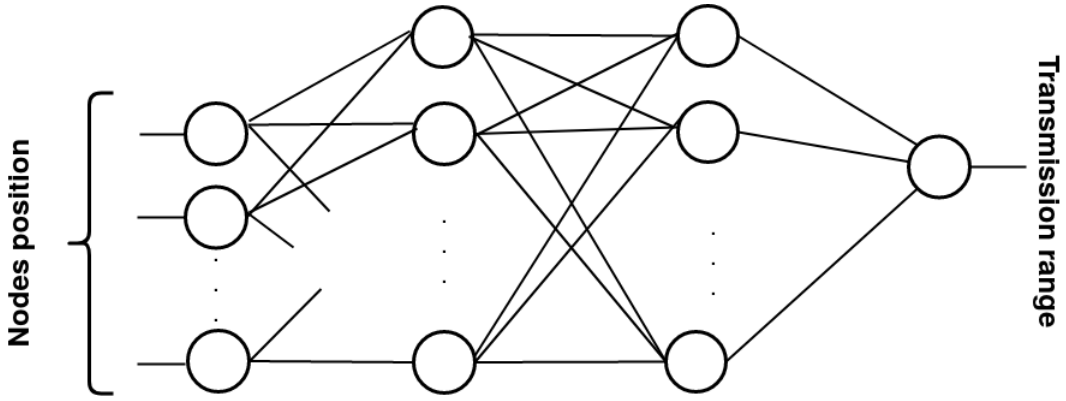


Figure 3.4 – Neural network architecture for efficient transmission range prediction.

the following metrics: the total energy consumption of the RPL network and the accuracy of the learning model.

Evaluation of the neural network model

To evaluate the performance of our learning model, we have considered three metrics from the WEKA tool: the Mean Absolute Error (MAE), the Correlation Coefficient (CC), and the Root Mean Square Error (RMSE). The MAE metric measures the average difference between the original values and the predicted values The WEKA's tool CC metric indicates the correlation between the predicted value and the original value. In order to have a perfectly correlated set of predictions, the value of the coefficient of correlation should be 1 otherwise the value 0 indicates a worst predictive model. The RMSE represents the average amount of errors made on the test set in the unit of the output variables.

We report the performance of our learning model in Table 3.2. This table shows that our model is capable of predicting a good transmission range value based on our training dataset. In fact, the resulting CC is very close to 1, whereas the values of MAE and RMSE are relatively low compared to the minimum transmission range value (20 m) considered in this work.

Table 3.2 – Performances of our MLP model.

Correlation Coefficient	0.9925
Mean Absolute Error	6.0684
Root Mean Squared Error	6.5782

Energy efficiency and network connectivity

We compared the total energy consumption generated by the default transmission range with the energy consumption generated with the estimated transmission

range. For some simulated RPL network topologies, we found that our model is able to estimate the transmission that enables a reduction of the network energy consumption (cf. Table 3.3) while ensuring the convergence of the RPL DODAG (cf. Figure 3.5).

Table 3.3 – Energy consumption for the RPL network topology T2.

	default transmission range (50 m)	**Estimated transmission range (72.03 m)**
Energy consumed (μAh)	1229	1059

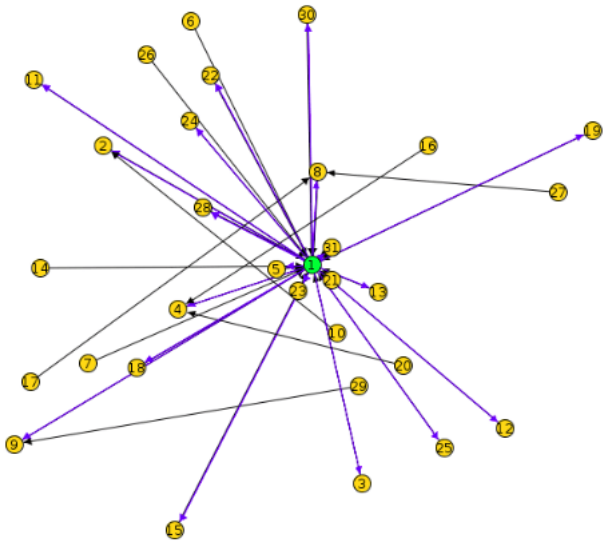


Figure 3.5 – RPL DODAG resulting from the predicted transmission range for the RPL network topology T2.

Discussion

As noted in above section, our proposed MLP model is able to determine the efficient transmission range for static RPL networks. However this solution presents some limitation:

- In one of the five topologies we used for testing, the estimated transmission range has generated a disconnected DODAG (some network nodes are isolated so that they cannot join the RPL DODAG) as show in Figure 3.6. This might be due to the fact that our training dataset does not contain enough data.

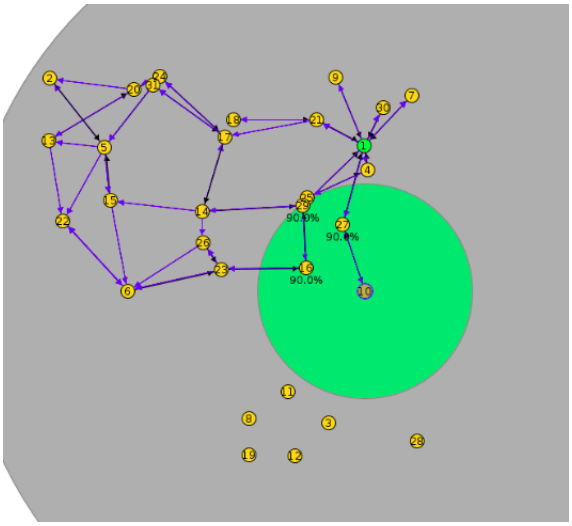


Figure 3.6 – RPL DODAG resulting from the predicted transmission range for the network topology T3.

- In a realistic deployment scenario, RPL networks are generally dynamic (i.e node can be added or removed to the network).

Considering these limitations, we propose an extension of our proposal in order to estimate the efficient transmission range of dynamic RPL networks. We discuss about that in the next Section.

3.4 Efficient transmission range for dynamic RPL networks

In this section, we provide a description of our solution that allows to determine the efficient transmission range for dynamic RPL networks. Figure 3.7 presents a high level view on the differents steps involved in our proposal.

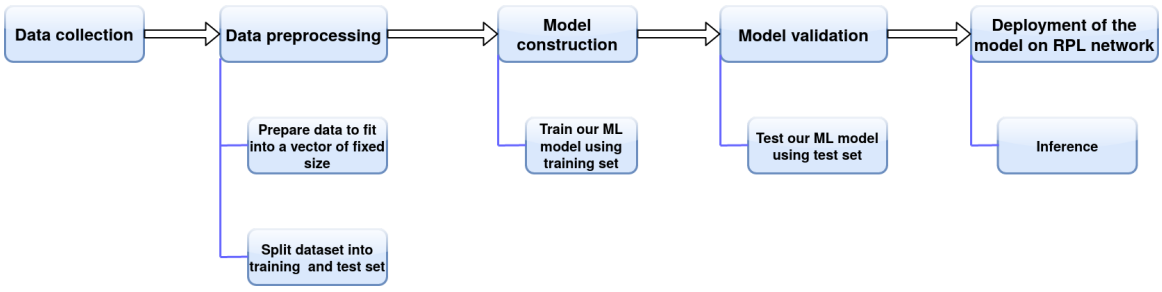


Figure 3.7 – Workflow of our proposal.

3.4.1 Problem statement

Let N_{max} and N_{min} be respectively the maximal and the minimal number of nodes of an RPL network over a network lifetime L. Let N_t denotes the number of nodes of an RPL network at time t such that $N_t \in [N_{min}, N_{max}]$. Let T, be the set that contains the position (the coordinates) of each node for a given RPL network topology deployed in an area of interest. The prediction of the efficient transmission range for a given topology is defined as solving a function y through historical measures of the efficient transmission range for a set of RPL network topologies. Nevertheless, IoT application domains such as smart cities and smart manufacturing, are characterized by RPL networks composed of resource-constrained devices whose number varies over the time due to node failure or node insertion into the network. For such IoT networks, our solution presented in section 3.3 cannot be used since its limited to scenarios where RPL networks size is fixed. Thereby, the main challenge here is to determine the efficient transmission range that improves the performance of an RPL network with a varying number of nodes.

3.4.2 Scenario A: Adaptive transmission range for dynamic RPL networks deployed in a 2D environment

3.4.2.1 Assumptions

We consider an IoT low power network with N_t static nodes deployed on a two-dimensional square field of $X \times Y$ surface area, with a stationary base station located at the coordinates $\left(\frac{X}{2}, \frac{Y}{2}\right)$. Moreover, we assume that all nodes are homogeneous in terms of wireless communication technology, computational capabilities, transmission power and have the same amount of initial energy on network setup.

In addition, we adopt the following model for total energy consumption of nodes:

$$E_{consumed} = \sum_{t=1}^{L} \left(E_A(t) + E_L(t) + E_{Rx}(t) + E_{Tx}(t)\right) \tag{3.3}$$

Where E_A refers to the energy consumed in active state over a period of time t, E_L refers to the energy consumed in low power CPU state over a period of time t, E_{Rx} refers to the energy spent in receiving state over a period of time t and E_{Tx} represents the energy consumed in the transmission state over a period of time t.

To generate the training dataset, we performed 10500 simulations using 1500 different network topologies (in terms of number of nodes and/or nodes position). For each topology, we varied the transmission range R, such that $R \in \{40, 50, \cdots, 100\}$.

Similarly to [153], we assume that the transmission power increases with the distance.

With the above considerations, we use the version 3.0 of the network simulator Cooja of Contiki [115] in order to obtain the transmission range that maximizes the network lifetime while maintaining the network connectivity for a set of RPL network topologies with various sizes.

3.4.2.2 Description of our solution

Since our proposal is based on a Supervised Learning algorithm, a set of data for the learning phase should be provided. For this purpose, we proceed to the collection of a historical set of efficient transmission ranges (the maximal distance to which a node can send its data to another one while minimizing the network energy consumption and preserving the network connectivity) and their associated network topology. To do this, we firstly generate a set of random RPL network topologies, with various sizes. During the topology generation phase, a minimum distance d_{min} between each two neighboring nodes is fulfilled. This minimum distance is used to guarantee that $\forall R \in \{40, 50, \cdots, 100\}$, the resulting RPL DODAG will be fully connected.

The minimum distance d_{min} can be formulated as follows:

$$d_{min} = \frac{\beta}{\sqrt{n}} \tag{3.4}$$

Where β is a parameter that we determine experimentally by considering n network nodes homogeneously spread over a square surface as depicted in Figure 3.8.

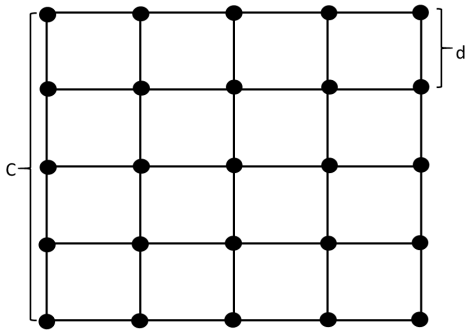


Figure 3.8 – Homogeneous deployment of nodes on a square surface.

From the Figure 3.8, we can deduce that the distance d between two neighboring nodes is:

$$d = \frac{C}{\sqrt{n} - 1} \tag{3.5}$$

Where C is the length of the side of the square and $n > 1$. From (3.5), we can deduce that:

$$d = \frac{C}{\sqrt{n} - 1} > \frac{C}{\sqrt{n}} \tag{3.6}$$

However, for each node, we know that $C > R_{max}$ (R_{max} is the maximal transmission range of a given node). Hence, from (3.6) we can deduce that:

$$d > \frac{C}{\sqrt{n}} > \frac{R_{max}}{\sqrt{n}} \tag{3.7}$$

Where $R_{max} \in]0, C[$.

Likewise, $\forall \beta \in]0, R_{max}[$, we deduce that:

$$d > \frac{R_{max}}{\sqrt{n}} > \frac{\beta}{\sqrt{n}} \tag{3.8}$$

Therefore, we chose the minimal distance between two neighboring nodes d_{min} ($d_{min} < d$) as expressed in (3.4).

Figure 3.9 shows an example of a topology generated with 100 nodes based on the d_{min} constraint.

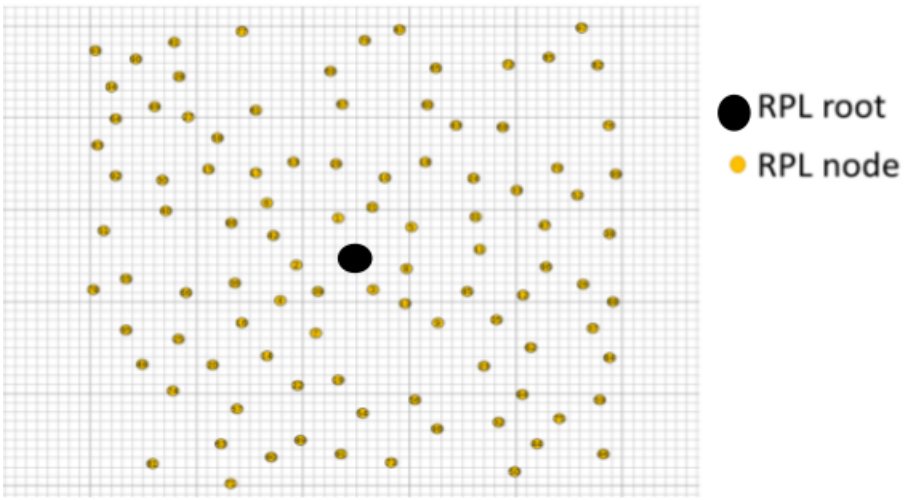


Figure 3.9 – Example of a generated topology with 100 nodes.

The pseudo-code depicted in Algorithm 3.2 illustrates the generation process of N topologies per number of nodes. For the sake of experimentation, we choose $N = 100$ and vary the number of nodes from 30 to 100 with a step of 5. The variation of the number of nodes allows to obtain a set of data composed with different network topology (in term of size and/or number of nodes)

After the generation of all the topologies, we used the Cooja simulator with the parameters depicted in Table 3.4 and performed the simulation of all the generated

Algorithm 3.2: Topologies generation

$N \leftarrow 100$
$totalNodesNumber = 100$
$X \leftarrow 500$
$Y \leftarrow 500$
$x_0 \leftarrow X/2$
$y_0 \leftarrow Y/2$
$D_{max} = \alpha$
for $k \leftarrow 30$; $k \leq nodesNumber$; $k = k + 5$ **do**
  for $m \leftarrow 1$ to k **do**
    for $i \leftarrow 1$ to N **do**
      $T_{N\_nodeNumber}$.add(x_0,y_0)
      generate (x_i,y_i)
      $check \leftarrow 0$
      compute d_{min} according to Equation (3.4)
      $j \leftarrow i$
      while $j \neq 0$ **do**
        $j \leftarrow j - 1$
        compute $D_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
        if ($D_{i,j} < D_{max}$) and ($d_{min} < D_{i,j}$) **then**
          $check++$
        else if ($D_{i,j} < d_{min}$) **then**
          generate (x_i,y_i)
          $j \leftarrow i$
        else if ($check == 0$) and ($j == 0$) **then**
          generate (x_i,y_i)
          $j \leftarrow i$
        end if
      end while
      $T_{N\_nodeNumber}$.add(x_i,y_i)
    end for
    $topologies$.add($T_{N\_nodeNumber}$)
  end for
end for
return *topologies*

network topologies. Since we need to evaluate the efficient transmission range of each topology, we incrementally varied the transmission range R from 40 to 100 with a step of 10. Once the full simulation completed, we collected the position of each node and the associated efficient transmission range (the transmission range that minimize the network energy consumption while ensuring the network connectivity). In total, we performed 10500 simulations on an OpenStack virtual machine comprising 24 CPUs running at 2.299 GHz each. The RAM size of the virtual machine is 16 GB and each simulation lasts 1200s. The pseudo-code of the data collection is shown in Algorithm 3.3, which is illustrated hereafter.

Algorithm 3.3: Data collection

$n \leftarrow 100$
$P = [40, 50, ..., 100]$
$nodeNumber = [30, 35, ..., 100]$
generate topologies for *nodeNumber* nodes using Algorithm 1
T.add(topologies)
for t_i in T **do**
  $bestTransmissionRange \leftarrow P_i$
  $DefaultEnergyConsumption \leftarrow \theta_i$
  for P_j in P **do**
    if (all motes have parent) **then**
      if ($E_{consumed} < DefaultEnergyConsumption$) **then**
        $bestTransmissionRange \leftarrow P_j$
        $DefaultEnergyConsumption \leftarrow E_{consumed}$
      end if
    end if
  end for
  dataset.add($t_i, bestTransmissionRange$)
end for
return *dataset*

Table 3.4 – Simulation parameters.

Parameter	**value**
Deployment area (m^2)	500×500
Number of nodes	30,35,40,45,50,55,60,65,70,75,80,85,90,95,100
Initial Energy(μAh)	1000000
Packets rate	1 packet/60s
Simulation time (ms)	1200000
Successful transmission ratio (Tx)	1.0
Successful receiving ratio (Rx)	0.9
Transmission range (m) (Tx)	40,50,60,70,80,90,100

Since our model aims to predict the value of the efficient transmission range of

an RPL network with a variable size, while the MLP inputs should be fixed, the length of the inputs of the MLP should be adapted so as to handle inputs of different sizes. For that purpose, we fixed L_{max} as the length of the input vector V of our MLP model. Thereby, during the training or testing steps, when the input vector length is shorter than L_{max}, the shorter input is padded to L_{max}. In other words, zeros are added to the input vector in order to get an input vector of size L_{max}.

Based on the collected dataset, we used Keras [34] and scikit-learn [120] libraries to build and train machine learning models.

3.4.2.3 Performance evaluation

To quantitatively assess the overall performance of our MLP model, we considered the Mean Absolute Error (MAE), the Mean Square Error (MSE) and the Coefficient of determination (R^2) in order to estimate the prediction accuracy. The MAE metric measures the average difference between the original values and the predicted values:

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j| \tag{3.9}$$

Where y_j is the original value, $\hat{y}_j$ is the predicted value and n represents the total number of predictions.

MSE is a metric corresponding to the expected value of the squared (quadratic) error or loss and it is defined as follows [120]:

$$MSE = \frac{1}{n}\sum_{i=0}^{n-1}(y - \hat{y}_i)^2 \tag{3.10}$$

R^2 is a metric which indicates how well the regression predictions approximate the real data points. Its best value is 1 and it is defined as follows [43]:

$$R^2 = 1 - \frac{\sum_{i=0}^{n-1}(\hat{y}_i - y_i)^2}{\sum_{i=0}^{n-1}(y_i - y)^2} \tag{3.11}$$

In addition, we performed test of our model on some RPL network topologies in order to assess the energy efficiency of our model compared to other machine learning models.

Prediction accuracy evaluation

To evaluate our predictive model that enables to estimate the efficient transmission range for RPL networks, we randomly splitted our dataset into two subsets:

- M_{train} : training set, which is composed of 80% of the data from our collected dataset.

- M_{test} : testing set, which is composed of 20% of the data from our collected dataset.

MLP has many parameters called hyperparameters that need to be defined before the training. Two possible methods can be used to define these hyperparameters. The first method consists in manually finding the good values of hyperparameters and then combining them. Nevertheless, this approach is not straightforward and can be an exhausting endeavor since it is based on manual search of the better combination of hyperparameters value over a large space of parameters. The second method is based on search algorithms such as Grid search, Random search, Simulated annealing and Tree-structured Parzen Estimators (TPE) [22].

We used an automated algorithm for hyperparameters optimization since it can rapidly provide the best parameters compared to the manual approach. However, the choice of an algorithm for hyperparameters optimization depends on the machine learning problem. Nonetheless, when the number of hyperparameters increases, the grid search algorithm becomes time consuming and computationally expensive. Thereby, we chose to evaluate the accuracy of the TPE, the random search and the Anneal search in order to select an algorithm for hyperparameters optimization. For this purpose, we used hyperopt library [90] to implement those algorithms. In addition, we run those algorithms using the same set of hyperparameters. The results of comparison of those algorithms are depicted in Figure 3.10. TPE turns out to be significantly more efficient than random search and anneal search. In fact, we observed that the validation accuracy of the TPE increases rapidly as the number of iterations increases compared to random search and anneal search. Therefore, we chose the best parameters provided by the TPE to build our model. The values of the hyperparameters of our model are depicted in Table 3.5.

However, since the size of the training set may impact the performance of the machine learning model, we evaluated the impact of the training set size on the accuracy of our model. Therefore, we performed several trainings on datasets with 400, 600, 800, 1000, 1200 and 1400 entries. We created a box and whisker plot (cf. Figure 3.11) to show the distribution of the test accuracy for each trained dataset. As depicted in Figure 3.11, we observed that the box and whisker plot shows that the dataset with 1000 instances provides a good spread of accuracy compared to other dataset. Hence, our M_{train} is composed of 1000 instances of the generated dataset.

We trained our MLP using the M_{train}. As shown in Figure 3.12, we observed

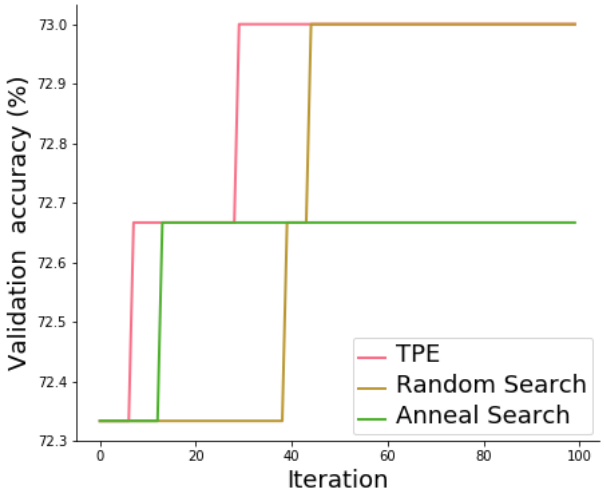


Figure 3.10 – Comparison of the accuracy of the TPE, random search and anneal search.

Table 3.5 – MLP parameters.

Parameter	value
Type of activation function	relu, linear
Loss function	MAE
Batch size	70
Number of epochs	1200
Training optimization algorithm	Adamax
Dropout regularization	0.5
Number of the hidden layer	4
Number of neuron in hidden layer	600

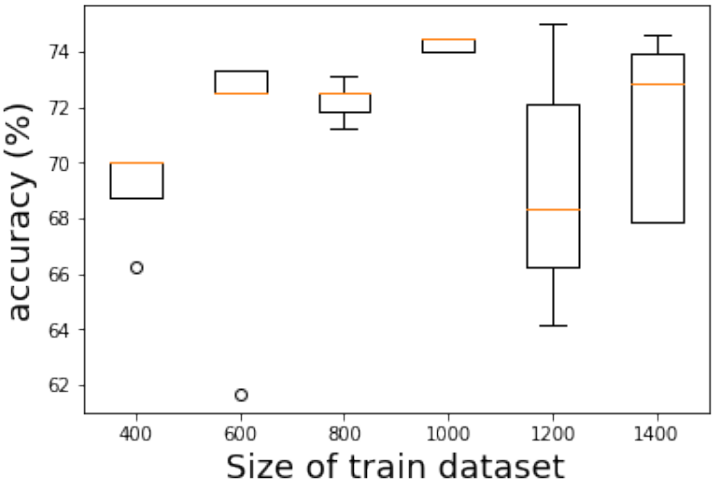


Figure 3.11 – Box and Whister plot of the distribution of test set accuracy of the MLP trained on different test set sizes.

that the loss of both training and validation decreases as the number of epochs (one forward and backward pass on all the training set) increases. This indicates that our model gains a high degree of accuracy as the number of epochs increases.

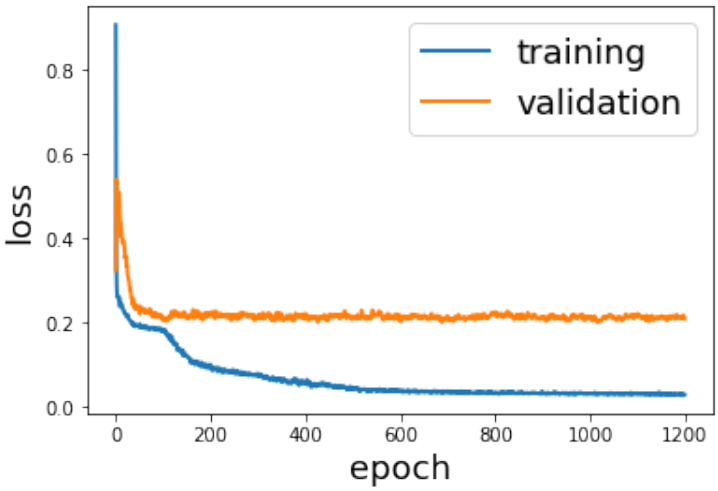


Figure 3.12 – Plot of Train and Test Loss of our MLP model.

In order to compare the MAE, MSE and R^2 of our MLP with other existing machine learning algorithms, we splitted our dataset obtained through simulation into k-folds. We used each fold once for validation and the remaining folds for the training. In our experimentation, we set the value of k to 4. Figure 3.13 shows the spread of the MAE accross each cross validation fold for each algorithm. This shows that our MLP model exhibits good performance compared to other machine learning models. Figure 3.14 shows the spread of the MSE accross each cross validation fold for each algorithm. In this case, the support vector regression exhibits good performance compared to other machine learning models. Figure 3.15 shows the spread of the R^2 score accross each cross validation fold for each algorithm. From that Figure, we can observe that MLP performs better compared other machine learning models.

Since MAE is a robust metric compared to MSE [29], we can conclude that our MLP model outperforms the other machine learning models.

Energy efficiency evaluation

We used the transmission range obtained through the different machine learning models (cf. Table 3.6) to evaluate the energy consumption of five RPL network topologies. As shown in Figure 3.16, it is not always possible to obtain a minimal energy consumption with our MLP model. This can be explained by the fact that we used padding technique in order to adapt the input of our MLP model so that

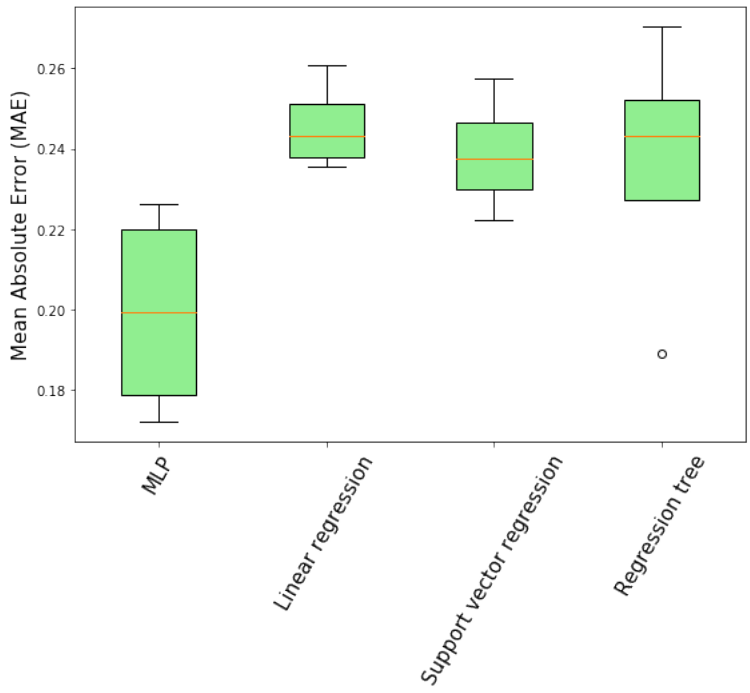


Figure 3.13 – Comparison of the MAE of different machine learning models.

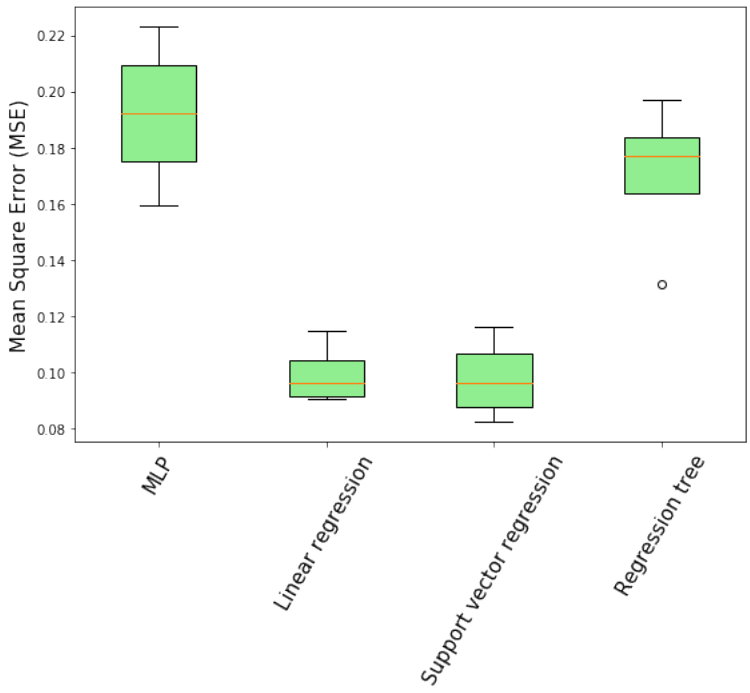


Figure 3.14 – Comparison of the MSE of different machine learning models.

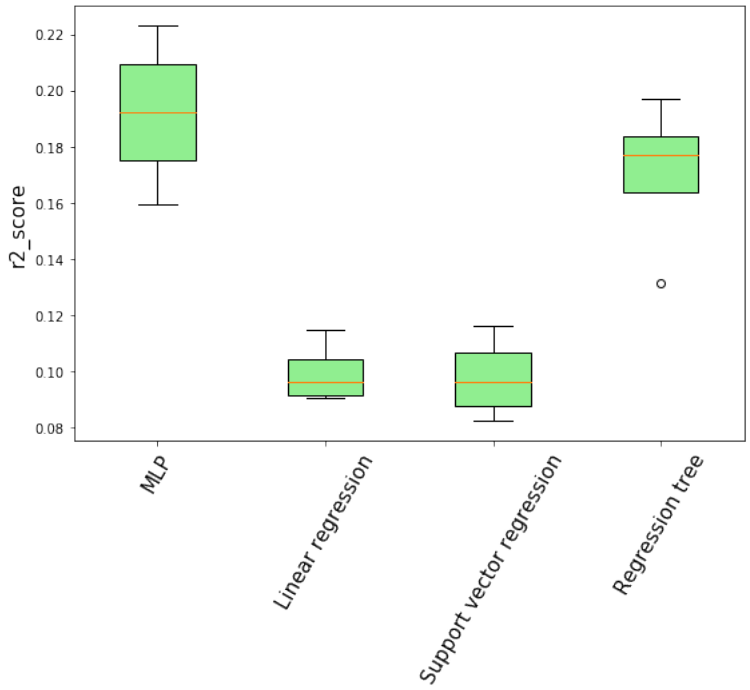


Figure 3.15 – Comparison of the R^2 score of different machine learning models.

it can handle dynamic RPL networks. In fact, as shown by the authors in [45], this technique may impact the performance of a given neural network.

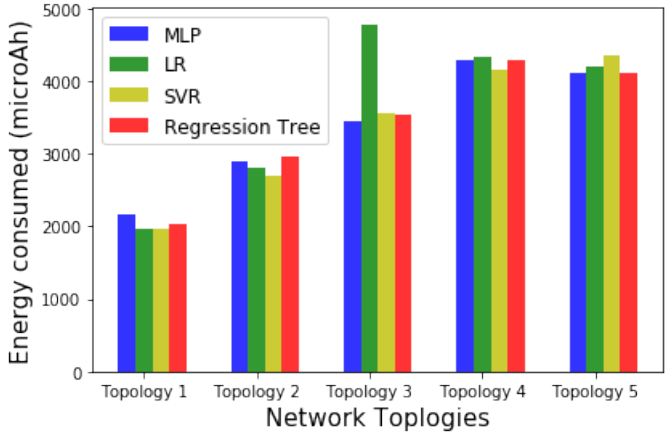


Figure 3.16 – Energy consumption generated by the tranmission range estimated by different machine learning models.

Table 3.6 – Transmission range values estimated by different machine learning models (in m).

Model	Topology 1	Topology 2	Topology 3	Topology 4	Topology 5
MLP function	86.07	86.05	94.18	94.15	100.00
Linear Regression	90.34	87.77	95.53	95.86	96.61
Support vector regression	93.11	94.84	96.40	97.34	97.57
Regression Tree	89.99	99.99	79.99	99.99	99.99

3.4.3 Scenario B: Adaptive transmission range for dynamic RPL networks deployed in a 3D environment

3.4.3.1 Assumptions

We consider an IoT low power network with N_t static nodes deployed in a three-dimensional field of $X \times Y \times Z$ surface area, with a stationary base station located at the coordinates $\left(\frac{X}{2}, \frac{Y}{2}, \frac{Z}{2}\right)$. Likewise, we assume that all the nodes are homogeneous in terms of wireless communication technology, computational capabilities, transmission power and have the same amount of initial energy on network setup. For the energy model, we adopt the same as in section 3.4.2.1.

3.4.3.2 Description of our solution

In a real deployment environment, IoT low power devices are generally deployed following a 3D topology and their number may vary over the time due to device failure, addition or removal of devices to/from the network and so forth. These events lead to an update of the network topology. Our proposal aims at providing a predictive model that determines the efficient transmission range of dynamic RPL networks deployed following a 3D environment. In the first step of our solution, a set of historical data containing 3D topologies and their efficient transmission range is collected. Then, we proceeded to extensive simulations of various network topologies in an $X \times Y \times Z$ area on Cooja simulation tool [115] where $X = 100m$, $Y = 50m$ and $Z = 200m$. The difference among the simulated topologies lies in nodes positions (coordinates) and the network size. An example of the configuration of a given node deployed following a 3D topology in Cooja is given in Figure 3.17.

For a given network topology, we varied the transmission range values in order to obtain the one that minimizes the network energy consumption while maintaining

```
<mote>
 <breakpoints />
 <interface_config>
  org.contikios.cooja.interfaces.Position
  <x>7</x>
  <y>23 </y>
  <z>63</z>
  </interface_config>
  <interface_config>
  org.contikios.cooja.mspmote.interfaces.MspClock
  <deviation>1.0</deviation>
  </interface_config>
  <interface_config>
  org.contikios.cooja.mspmote.interfaces.MspMoteID
  <id>1</id>
  </interface_config>
  <motetype_identifier>z12</motetype_identifier>
</mote>
```

Figure 3.17 – Example of configuration of Cooja node deployed in 3D environment.

the network connectivity. We repeat this process for various network topologies sizes. In our simulation, we varied the number of devices from 30 to 100 with a step of 5. For each simulated RPL network, we incrementally varied the transmission range from 40 to 100 with a step of 10. In total, we performed 10500 simulations on an OpenStack virtual machine comprising 24 CPUs running at 2.299 GHz each. The RAM size of the virtual machine is16 GB and each simulation lasts 1800s. At the end of the simulation, we obtained a dataset composed of 1500 entries. Each entry is composed of the efficient transmission range value and the associated RPL network topology.

After the collection of the historical data composed of 3D topologies and their efficient transmission range, we applied the same techniques as in our model that determines the efficient transmission range in dynamic RPL networks deployed in 2D environment. In fact, we applied padding technique to allow the MLP consider dynamic RPL networks. Additionally, we used TPE in order to optimize the hyperparameters of our model. The values of those hyperparameters are depicted in Table 3.7

Table 3.7 – MLP parameters

Parameters	Value
Type of activation function	relu, linear
Loss function	MAE
Batch size	90
Number of epochs	800
Dropout regularization	0.5
Number of the hidden layer	3
Number of neuron in hidden layer	300
Training optimization algorithm	Adam

3.4.3.3 Performance evaluation

To evaluate the performance of our MLP model, we considered the Mean Absolute Error (MAE) and the Mean Square Error metrics. Furthermore, we also compared the energy consumption generated using our model to the energy consumption generated using other state of the art machine learning models. We evaluated the perfomance of our MLP model in three steps:

Step 1

We splitted our dataset into k-fold and used each fold once for validation and the remaing folds for the training. In our experimentation, we set the value of k to 6. We compared the performance of our MLP model with some state of the the art machine learning models based on the same training and validation set. Figure 3.18 and 3.19 shows the spread of the MAE and MSE accross each cross validation fold for each algorithm. These results show that our MLP exhibits a good performance compared to other machine learning models such as linear regression, support vector regression and regression tree. This proves that our MLP model can estimate well the efficient transmission range compared to other machine learning models.

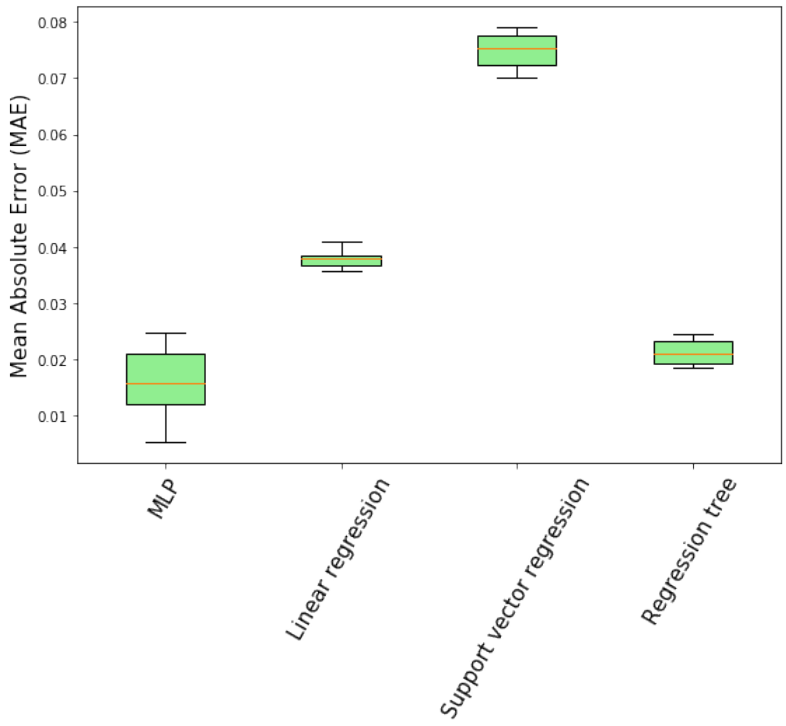


Figure 3.18 – Comparison of the Mean Absolute Error of different machine models.

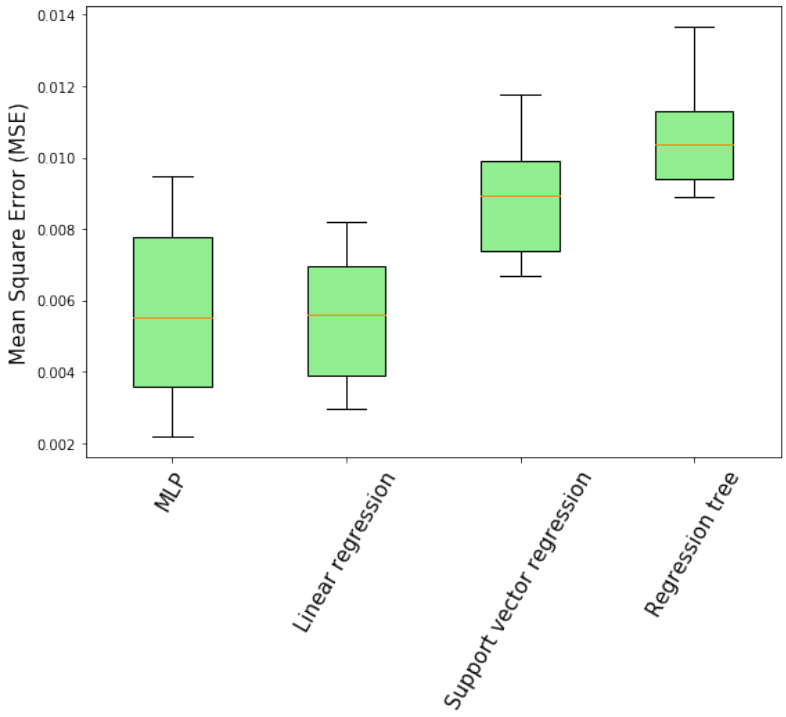


Figure 3.19 – Comparison of the Mean Square Error of different machine models.

Step 2

We randomly splitted our dataset obtained through simulation in two subsets as follows: 80 % for training and 20 % for the test. After that, we reduced successively 10% and 20% of the output of our training set in order to obtain a dataset with missing values. By doing so, we can observe the performance of our MLP model when using a dataset with a reduced size during the training. In fact, for a given machine learning model, the size of the training set may impact the performance of the model. Additionally, in real world it is not straighforward to create 10k network topologies and identify their efficient transmission range as we did during simulations.

We employed two strategies to deal with the remaining training set. The first strategy is to predict the missing output values using the linear regression. In the second strategy, we removed the entries containing the missing output. We obtained different types of training dataset namely: D1 and D4, the training set composed respectively of 10% and 20% of predicted values; D2 and D5, the training set composed respectively of 10% and 20% of deleted rows. D3 is the training set without missing values. We trained different MLP models using those training dataset and the hyperparameters given in Table 3.7. The results given in Table 3.8 show that models based on the training set with missing rows provides MAE values close to the MAE of the model based on training set without missing values.

This means that our model can provide a good performance in a real world with a reduced training dataset.

Table 3.8 – Comparison of the MAE of MLP for different training sets.

Training set	MAE
D1 (dataset with 10 % of predicted values)	0.10
D4 (dataset with 20 % of predicted values)	0.10
D3 (dataset without missing values)	0.02
D2 (dataset with 10 % of deleted rows)	0.01
D5 (dataset with 20 % of deleted rows)	0.02

Step 3

Concerning the energy efficiency of our model, we performed various simulations of a number of network topologies on Cooja using the transmission range estimated by the different machine learning models. Each simulation lasted 1800 s and we evaluated the network energy consumption generated when using the transmission range estimated using the different machine learning models. Figure 3.20 shows that it is not always possible to obtain a minimal energy consumption using our MLP model. This limitation is due to the error (difference between the predicted and the real value) obtained during the training of our model.

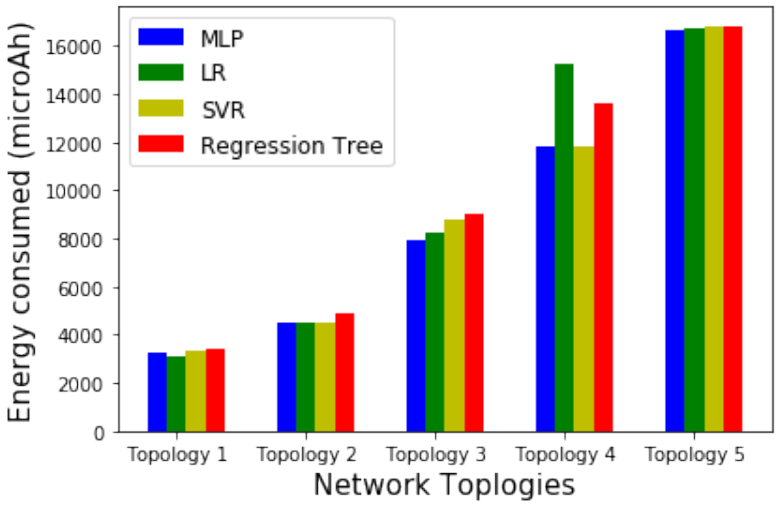


Figure 3.20 – Comparison of the network energy consumption generated by the transmission range of different machine learning models.

3.5 Deployability of our solution

Our solution can be deployed in an RPL network managed by a central entity (e.g. the RPL root) (cf. Figure 3.21). We assumed that this entity is a node with a powerful computing capacity so that the training of our model can be performed by that entity. In order to enable the update of the value of transmission range in RPL network, the network manager should be aware of the network topology. For that purpose, each node in the network notifies its presence to the network manager through a message containing the position of the node. After that, the network manager will compute the efficient transmission range by using our ML model and then trigger the update of the transmission range value of nodes in the RPL network. Figure 3.22 shows the process of topology discovery and the update of nodes transmission range value.

Algorithm 3.4 shows an overview of the process executed by the network manager in order to update the transmission range of an RPL network. The process executed by each node is summarized in algorithm 3.5. Each RPL node is responsible for sending its position to the network manager so that it can receive and update of its transmission range.

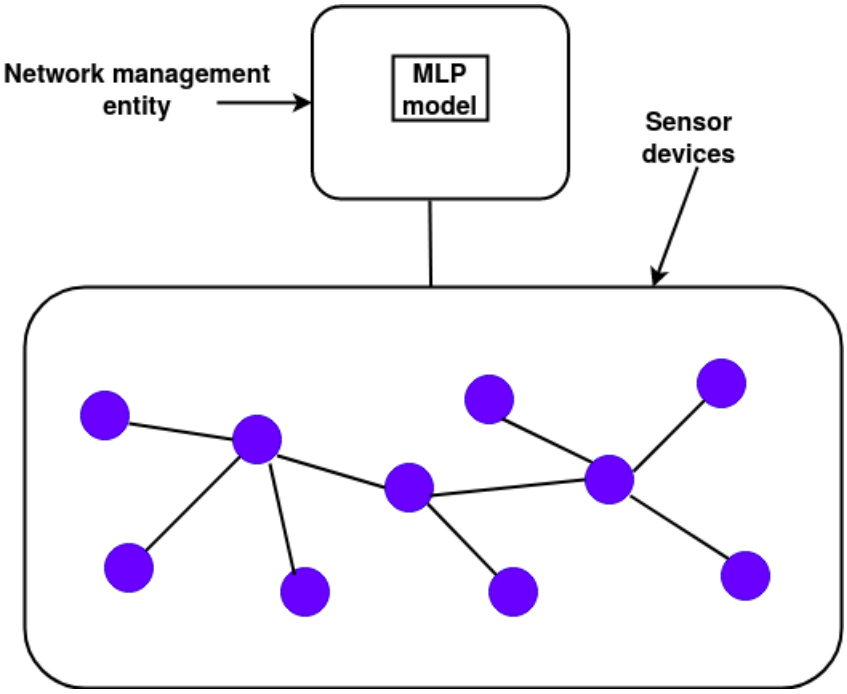


Figure 3.21 – High-level illustration of deployment of our solution in IoT networks.

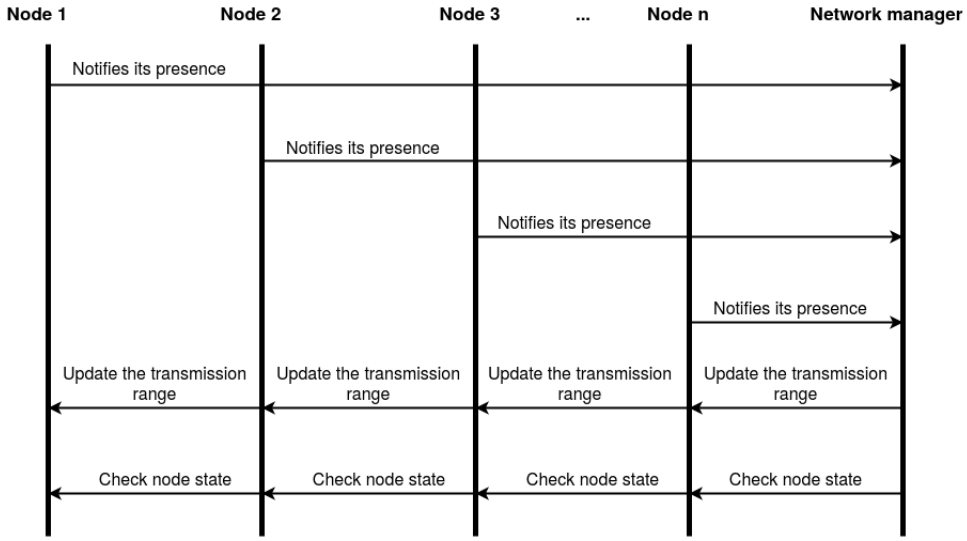


Figure 3.22 – Topology discovery.

Algorithm 3.4: Process at the network manager

```
1: build MLP model using collected data
2: wait for a network topology (∗)
3: if (new topology is received) then
4:    use MLP model to estimate the efficient transmission range
5:    send the efficient transmission range to network devices
6:    go back to (∗)
7: end if
```

Algorithm 3.5: Process at the network device

```
1: while energy of device is not exhausted do
2:    send its position to the network manager (∗)
3:    wait for the efficient transmission range
4:    if transmission range is received then
5:       update the transmission range
6:    end if
7:    if device position is updated then
8:       go back to (∗)
9:    else
10:      wait device position update
11:      go back to (∗)
12:   end if
13: end while
```

3.6 Conclusion

In this chapter, we investigated the problem of determining the efficient transmission power of RPL networks and then, we proposed intelligent solutions based on MLP in order to estimate the efficient transmission range in those IoT low power networks. Our proposed solutions consider RPL nodes positions in order to determine the transmission range that enables a reduction of the network energy consumption while maintaining the network connectivity. The results of comparison show that our solutions outperform other existing machine learning models such as linear regression, support vector regression and regression tree. These solutions have been published in [3, 4].

Efficient configuration of IEEE 802.15.4 MAC

Sommaire

4.1 Introduction

Owing to the limitation of IoT low power networks (e.g. low memory and limited CPU of IoT devices), the optimization of their performance has been extensively studied in the literature [75, 118]. In those networks, the desired performances metrics are: low end-to-end delay, low packet loss and high throughput with low energy consumption. These requirements are critical for IoT applications such as forest fire detection, target tracking and warehouse tracking [27]. To achieve that, it is necessary to select the appropriate protocols. In IoT low power networks, the IEEE 802.15.4 standard has been widely used in various solutions [60] (e.g. Thread, Zigbee, 6LoWPAN) and proved to be suitable for resource-constrained networks.

The IEEE 802.15.4 standard defines the MAC layer and physical layer, which allow to access to the wireless medium for the transmission of MAC frames in Low-power and Lossy Networks. Due to the resource constraints of IoT devices, it has been shown that it is necessary to select the appropriate MAC parameters in order to satisfy the performance requirements for IoT applications [116]. In the literature, various approaches that enable the definition of IEEE 802.15.4 MAC parameters have been proposed [127, 30]. Nonetheless, in dynamic IoT low power networks,

existing solutions are inefficient since they do not include mechanisms that help to dynamically adapt the MAC parameters according to network characteristics and network traffic conditions, hence optimizing the communication delay. To cope with this issue, machine learning techniques can be exploited [36, 8] in order to devise a solution that enables an efficient definition of values of IEEE 802.15.4 MAC parameters in dynamic IoT low power networks.

In this chapter, we present a novel solution based on a supervised learning algorithm, to define the values of IEEE 802.15.4 MAC parameters, according to the current network traffic conditions and the network characteristic. The proposed solution helps in improving the end-to-end network latency.

The rest of the chapter is organized as follows: section 4.2 presents background; section 4.3 presents an overview on existing solutions for configuring IEEE 802.15.4 MAC parameters; section 4.4 presents the proposed solution and the performance evaluation; section 4.5 concludes this chapter.

4.2 Background

The Media Access Control (MAC) and physical (PHY) layer for Low-Rate Wireless Personal Area Networks (LR-WPAN) are defined by the IEEE 802.15.4 standard [86]. The IEEE 802.15.4 MAC supports the Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA) mechanism for channel access [49]. This mechanism helps to avoid collision during the data transmission and it can operate in slotted or unslotted mode. The slotted mode works in beacon-enabled mode while the unslotted mode works in a non beacon-enabled mode. The unslotted mode is adapted for IoT low power networks as shown in [158]. We will focus on that mode in this work.

The performance of a network using CSMA-CA depends on the MAC parameters such as the Backoff exponent (BE), the Maximum CSMA backoff (macMaxCSMABackoffs) and the Maximum frame retries limit (macMaxFrameRetries) [10, 127]. In CSMA-CA a sender must sense the channel before packet transmission. If the channel is not free it must wait for a specified number of backoff periods determined by the variable Backoff Exponent (BE) (0 - $(2^{BE} - 1)$). The BE is initiated at 'macMinBE' value and it is doubled after each unsuccessful channel sensing until it reaches 'macMaxBE'. Once this value is reached, there is no more doubling of the BE value after unsuccessful channel sensing. The macMaxCSMABackoffs represents the number of times a given sensor device stays in the backoff stage after unsuccessful channel sensing before the packet is dropped. The macMaxFrameRetries represents the maximum number of retransmissions of

a paquet when there is no acknowledgement. The value of those parameters as suggested by the standard [30, 8] are given in Table 4.1.

Table 4.1 – IEEE 802.15.4 MAC parameters value.

Parameter	**Value Range**	**Default value**
macMinBE	0-7	3
macMaxBE	3-8	5
macMaxCSMABackoffs	0-5	4
macMaxFrameRetry	0-7	3

4.3 Overview on existing solutions

In order to improve the performance of IoT low power networks that use the IEEE 802.15.4 MAC, the authors in [89, 117, 30] have demonstrated that it is necessary to efficiently define the values of IEEE 802.15.4 MAC parameters to achieve a good network performance. In these works, they showed that the optimal values of parameters of IEEE 802.15.4 MAC depend on network traffic characteristics (e.g. sending frequency of data). In the literature, various method have been proposed to enable the selection of the set of IEEE 802.15.4 MAC parameters that allows to achieve a good network performance. In particular, due to dynamic nature of IoT low power networks (caused by the mobility of devices, the dynamic nature of wireless sensor networks and so forth), various approaches based on machine learning algorithms have been proposed to improve the performance of IEEE 802.15.4 MAC [24].

Alberola and Pesch [42] proposed a method based on a reinforcement learning algorithm (q-learning), in order to allow defining the duty cycle so as to achieve good network performance. In the same vein, Liu and Elhanany [99] proposed a novel MAC protocol that employs a reinforcement learning algorithm to dynamically adjust the duty cycle of sensor devices. The reinforcement learning algorithm helps to infer the state of other sensor devices in order to allow a given sensor node to consider both its own traffic load and the traffic load of its neighbouring to adjust its duty cycle. However, in these works, the authors do not consider the 802.15.4 MAC parameters such as Backoff exponent and the Maximum frame retries limit. Consequently, non-optimal MAC parameters may be used, leading to a network with a poor performance characterized by a high end-to-end delay for data transmission and an excessive network consumption.

Collotta et al. [36] proposed an approach based on fuzzy controller to allow a dynamic adjustement of the Contention Windows (CW) and the Backoff Exponent

according to the throughput and the workload of an industrial IEEE 802.15.4 wireless sensor network. However, in this work, the authors do not consider other MAC parameters such as the maximum frame retry, which may also impact the network perfomance.

AL-KASEEM et al. [8] proposed a method based on artificial neural networks and genetic algorithms to select the 6LowPAN MAC parameters (backoff exponent, maximum CSMA backoff and maximum frame retries limit) that lead to a good network performance. However, in the proposed method, the dataset used by the authors do not consider the network traffic characteristics which is also a key element to determine the optimal set of IEEE 802.15.4 MAC parameters.

Considering the limitation of the above works, we propose a novel scheme based on machine learning in order to dynamically define the optimal values of IEEE 802.15.4 MAC parameters by considering the network characteristic (the network density) and the characteristic of the network traffic (the sending rate frequency and the paquet interarrival time). Our solution aims at improving the end-to-end delay in IoT low power networks based on IEEE 802.15.4 MAC.

4.4 Context aware configuration of IEEE 802.15.4 MAC

4.4.1 Assumptions and problem formulation

Let us consider a wireless sensor networks composed of N_i devices randomly deployed in a disk area of radius 500 meters centered around a sink node, that is aimed at monitoring the parameters of an environment such as temperature, pollution and so forth. Let d_i, be the density (defined as the average number of neighbor of each node in the network) of a given network topology. Let f_i, be the average interval between sensing events at nodes level. Let t_i, be the average delay between packet arrivals at the sink node. N_i, d_i, f_i and t_i represents some characteristics of a WSN and they can be considered to define an optimal set of IEEE 802.15.4 MAC parameters. Due to the dynamic nature of WSNs (caused for example by the addition or failure of nodes), for a given wireless sensor network, an infinite possible values exists for N_i, d_i, f_i and t_i. Those values should be considered in order to define the optimal set of IEEE 802.15.4 MAC parameters. In the literature, machine learning techniques have been used to solve such kind of problem [53]. Nonetheless, their application in IoT low power networks remains a research challenge because of the inherent constraints of those networks (e.g. limitation in

terms of CPU, memory and battery).

In the context of network management, machine learning techniques are generally based on a set of metrics (e.g. routing metrics) that helps building a model that can be used to take decision such as routing path update or the reconfiguration of network device parameters. In this work, our goal is to build a model that can enable the reconfiguration the IEEE 802.15.4 MAC parameters based on the network and data traffic characteristics.

Let $S = \{N, D, F, T, P1, P2, P3, P4\}$, be the training dataset of k elements composed of network charasteristic as inputs and the optimal IEEE 802.15.4 MAC parameters as outputs. $N = \{N_1, \cdots, N_n\}$ represents the set that contains the number of nodes in the network; $D = \{d_1, \cdots, d_n\}$ represents the set that contains different network density; $F = \{f_1, \cdots, f_n\}$ represents the set that contains the values of average interval between sensing events at nodes level; $T = \{t_1, \cdots, t_n\}$ represents the set that contains the values of average delay between packet arrivals at the sink node. $P1 = \{p_1^1, \cdots, p_1^n\}$ represents the set that contains different value of parameter $p1$ (macMinBE). $P2 = \{p_2^1, \cdots, p_2^n\}$ represents the set that contains different value of parameter $p2$ (macMaxBE). $P3 = \{p_3^1, \cdots, p_3^n\}$ represents the set that contains different value of parameter $p3$ (macMaxCSMABackoffs). $P4 = \{p_4^1, \cdots, p_4^1\}$ represents the set that contains different value of parameter $p4$ (macMaxFrameRetry). The problem consists in learning a multi-target predictive model from S so that we can build a model that allows to predict the approriate values of parameters of IEEE 802.15.4 MAC.

4.4.2 Description of our solution

Before describing our proposal, we provide a description of the process of generation of our dataset for buiding a machine learning model.

Training data generation

We performed various simulations using a simulator (cf. Figure 4.1) for wireless sensor networks, which has been developed in LSC laboratory of the French Alternative Energies and Atomic Energy Commission (CEA). This simulator offers the possibility to generate random mesh network layouts deployed around a single root node or sink node. It is event driven rather than time driven for the sake of efficiency. Each node generates sensing events according a Poisson random process. This type of event translate then into packet transmission events either from the source node or from intermediate nodes toward the sink that behave as routing nodes. The simulator embeds a simple radio communication model which allows to

address coverage and packet collision issues. Standard IEEE 802.15.4 CSMA/CA state machines has been included in each node in order to address mechanisms such as packets collisions, packet acknowledgments, repetitions and so on. Finally, the simulator can be used in a graphic mode where the state of each node in the mesh network is shown according to color codes (cf. Figure 4.1). Moreover, it is also possible to run the simulator in non graphic scriptable mode.

During the generation of the training dataset, we varied the number of nodes in the network from 400 to 500 with a step of 1. For each network size, we have instanciated different network topologies and varied the average interval between two sensing. The event generation at each node is following the poisson distribution. Each simulation lasts x seconds and at the end of each one, we retain the inputs and outputs that allow both a reduction of the packet loss and the average end-to-end delay for transmission of paquets. At the end we obtained a dataset composed of 9090 entries. The pseudo code describing the process of generation of our training data is given in Algorithm 4.1.

Algorithm 4.1: Training data generation

$N \leftarrow [400, 401, ..., 500]$
$nDensity \leftarrow [d_1, i_2, ..., d_l]$
$MacParams \leftarrow [params_1, params_2, ..., params_j]$
$I \leftarrow [i_1, i_2, ..., i_m]$
$bestMacParams \leftarrow defaultMacParameters$
for $nNodes$ in N **do**
  for $density$ in $nDensity$ **do**
    for $interval$ in I **do**
      for $parameters$ in $MacParams$ **do**
        if (end-to-end delay is reduced) **then**
          if (packet loss is reduced) **then**
            $bestMacParams \leftarrow parameters$
          end if
        end if
      end for
      save($nNodes, density, interval, bestMacParams$)
    end for
  end for
end for

Intelligent (re)configuration of IEEE 802.15.4 MAC parameters

Our proposed mechanism for defining the optimal IEEE 802.15.4 MAC parameters is depicted in Figure 4.2. The proposed mechanism leverage the benefit of machine learning techniques in order to predict the values of IEEE 802.15.4 MAC

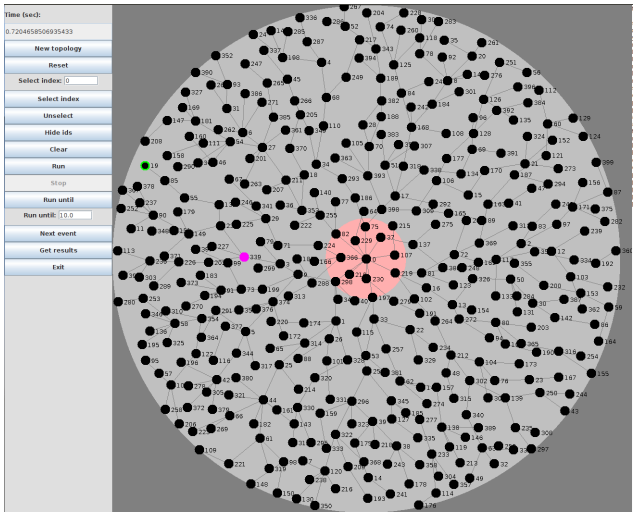


Figure 4.1 – Event-driven simulator for wireless sensor networks.

parameters that allows a reduction of the end-to-end delay. The configuration of the MAC parameters is performed during network startup and also when there is an update in the network.

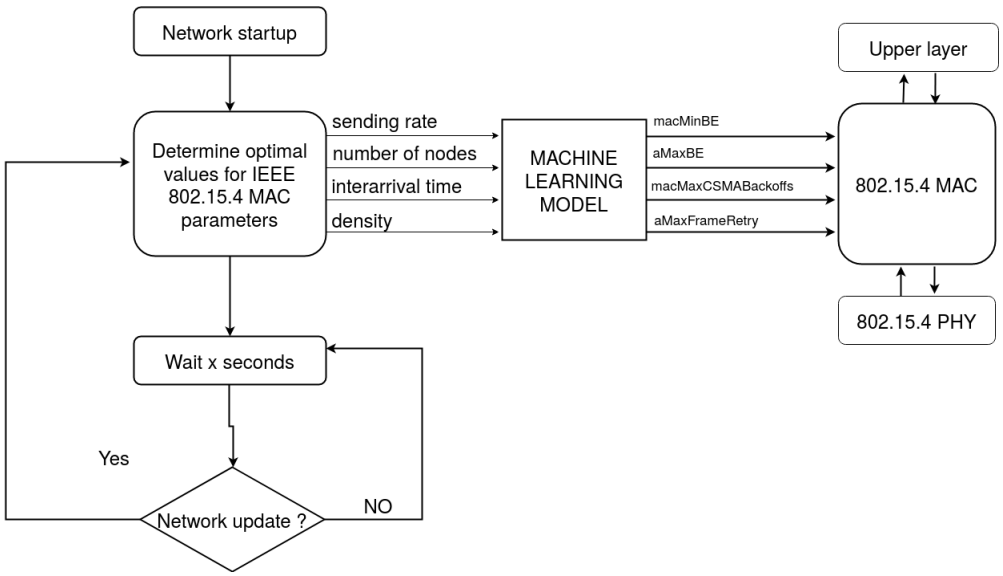


Figure 4.2 – Scheme for definition of optimal IEEE 802.15.4 MAC parameters using a machine learning model.

Based on the state of the art of machine learning techniques used in networking, we considered multilayer perceptron, random forest, K-nearest Neighbors and decision tree algorithms in order to build a predictive model that enables to determine the optimal parameters for IEEE 802.15.4 MAC.

A multilayer perceptron (MLP) is a type of neural network which has been used to perform classification and regression tasks based on a given set of data. We provided more details about MLP in section 3.2.2.

The decision tree is a type of machine learning algorithm which enables to perform classification or regression tasks based on a given set of historical data. Typically, a decision tree consists of root nodes, hidden nodes and a lot of terminal nodes or leaves. The implementation of decision tree classifier using scikit-learn library [119] includes various parameters, namely criterion, splitter, max_depth, min_samples_split, to name a few. In the context of wireless sensor networks, the decision tree algorithm has been used to perform tasks such as anomalies detection.

The K-nearest Neighbors classifier is a type of machine learning algorithm, which can be used for regression or classification tasks. The idea behind this classifier is to find distance (e.g. Euclidian distance, Minkowsky distance, Hamming distance) between the test data and each of the training data to determine the final output. The implementation of that classifier using scikit-learn [119] uses these parameters: n_neighbors, weights, algorithm, leaf_size and so forth. In the context of wireless sensor networks, the K-nearest Neighbors has been used for solving problems such as target location.

Random forest classifier corresponds to a collection of decision trees used to perfom machine learning tasks such as classification or regression. The classifier can be implemented using scikit-learn library [119] and it uses these parameters: n_-estimators, criterion, max_depth, min_samples_split and so forth. In the context of wireless sensor network, this classifier has been used for solving problem such as intrusion detection.

Figure 4.3 illustrates the process of construction of our learning model that enables to select the values of IEEE 802.15.4 MAC parameters. The dataset we collected is used to devise models that output each a value of IEEE 802.15.4 MAC parameters. However, given the fact that certain events do not occur frequently in WSNs, during the generation of our training dataset, it is likely to obtain an imbalanced set of data. Building a learning model based on imbalanced dataset may lead to obtaining a biased model that allows taking wrong decisions. This may have severe impact on the network performance. Thereby, to cope with this issue, for each model, we performed a resampling by using Borderline-SMOTE, an algorithm that helps to avoid having an imbalanced dataset [66]. By doing so, we obtained different datasets to devise models that help to determine the optimal values of IEEE 802.15.4 MAC parameters.

In order to select the model that perform well for determining the values of IEEE 802.15.4 MAC parameters in dynamic IoT low power networks, we trained the above classifier using the different training sample. We used TPE in order to search the best hyperparameters of those classifiers. The best values we obtained for those classifiers are depicted in Table 4.2, 4.3, 4.5 and 4.4 . We compared those models by

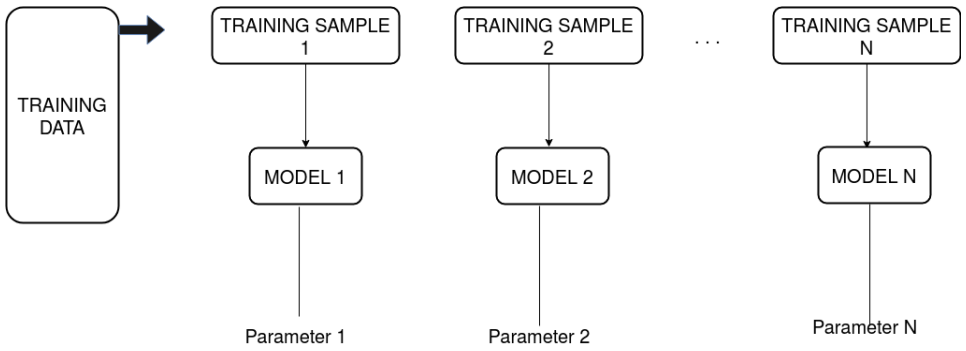


Figure 4.3 – Prediction of the values of IEEE 802.15.4 MAC parameters.

considering the Geometric mean, the precision, the accuracy and F-score metrics. These metrics are defined as follows [56]:

- $Geometric - mean$ $(G - mean)$: is a metric commonly used to evaluate a classifier trained using imbalanced data:

$$G - mean = \sqrt{Recall * specificity} \tag{4.1}$$

where: $specificity = \frac{TN}{TN+FP}$ and $recall = \frac{TP}{FN+TP}$. TP or True Positive corresponds to the amount of samples predicted as class C when they truly belong to class C. TN or True Negative corresponds to the amount of samples not belonging to C and not classified as C. FP or False Positive corresponds to the amount of samples not belonging to class C while classified as C. FN or False Negative corresponds to the amount of samples that are not classified as C while they belong to class C.

The best value of $G - mean$ is 1 and the worst value is 0.

- $Accuracy$: this metric corresponds to the ratio of the number of correctly classified instances to the total number of input samples. The formula of $Accuracy$ is:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.2}$$

- $Precision$: this metric corresponds to the percentage of examples that are correctly labeled as positive [56]. The formula of $Precision$ is:

$$Precision = \frac{TP}{TP + FP} \tag{4.3}$$

The best value of $Precision$ is 1 and the worst value is 0.

- $F - score$: this metric represents a weighted average of the precision and

recall. The best value of $F - score$ is 1 and worst score at 0. The formula of $F - measure$ is:

$$F = \frac{(1 + \beta^2) * precision * recall}{\beta^2 * precision + recall} \tag{4.4}$$

Table 4.2 – Best hyperparameters of Random Forest Classifier obtained with TPE.

Parameter	**Value**
max_depth	4000.0
max_features	log2
min_samples_leaf	2.0
min_samples_split	90.0

Table 4.3 – Best hyperparameters of KNeighbors Classifier obtained with TPE.

Parameter	**Value**
algorithm	brute
leaf_size	430.0
n_neighbors	65.0
weights	distance

Table 4.4 – Best hyperparameters of Decision Tree classifier obtained with TPE.

Parameter	**Value**
criterion	gini
max_depth	3010.0
max_features	None
min_samples_leaf	130.0
min_samples_split	202.0

Table 4.6 shows the comparison of the accuracy, precision, F-score and Geometric for different machine learning. Additionally to the comparison of those models based on the above metrics, we also evaluated the confidence interval of classification accuracy of those classifiers. As mentioned in [80], the confidence interval tells us how precise our estimate is likely to be. Table 4.7 shows the 90% confidence interval of classification accuracy of different machine learning models. It shows that Random forest and K-nearest Neighbors perform well compared to other machine learning classifiers since they provide the smaller confidence interval. Based on those results we chose the random forest in order to estimate the appropriate values of IEEE 802.15.4 MAC parameters.

Table 4.5 – Best hyperparameters of the Multilayer Perceptron Classifier obtained with TPE.

Parameter	**Value**
batch_size	350.0
learning_rate	invscaling
max_iter	300.0
number of hidden layer	1.0
number of neurons	56.0
solver	adam

Table 4.6 – Comparison of the performance of different machine learning models.

Model	**Precision**	**Accuracy**	**F-score**	**Geometric mean**
multioutput Decision Tree	0.4181	0.4765	0.4013	0.5362
multioutput Random Forest	**0.4656**	**0.4924**	**0.4275**	**0.5548**
multioutput K-nearest Neighbors	0.4523	0.4822	**0.4430**	**0.5591**
multioutpout Multilayer Perceptron	0.2676	0.4761	0.3206	0.5013

Table 4.7 – 90% confidence interval of classification accuracy of different machine learning models.

Model	**Interval**
multioutput Decision Tree	+/- 0.0523
multioutput Random Forest	+/- 0.0520
multioutput K-nearest Neighbors	+/- 0.0520
multioutpout Multilayer Perceptron	+/- 0.0522

4.4.3 Performance evaluation

To evaluate the performance of our proposed approach that aims at finding optimal values for IEEE 802.15.4 MAC parameters, we considered the average end-to-end delay metric. This metric corresponds to the average delay between packet generation in a source node and packet receiving in the sink node.

We analysed the end-to-end delay of our proposed scheme to determine the optimal values for IEEE 802.15.4 MAC parameters by varying the average interval between sensing at each node. Figure 4.4 shows the average end-to-end latency versus the average sensing interval for a sensor network composed of 400 devices. We observed that MAC parameters estimated using the random forest classifier can help achieving a better performance compared to the default parameters suggested by the IEEE 802.15.4.

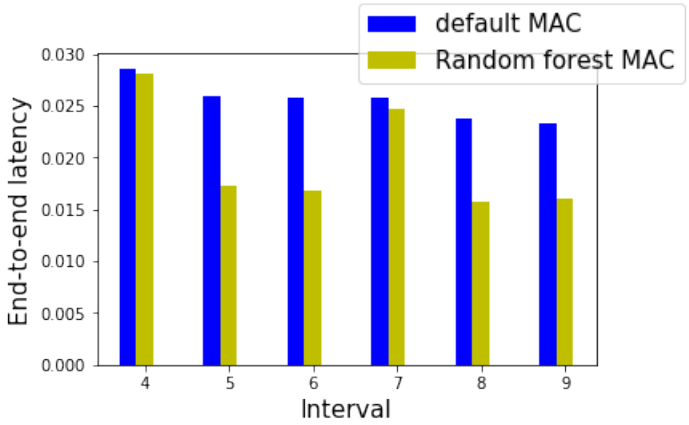


Figure 4.4 – End-to-end delay comparison.

4.5 Conclusion

In this chapter, we proposed an efficient and adaptative solution for defining the IEEE 802.15.4 MAC parameters of dynamic IoT low power networks, by taking into account the network and data traffic characteristics. In the proposed solution, the IEEE 802.15.4 MAC parameters are defined using a predictive machine learning model. The results of simulation show that the parameters of IEEE 802.15.4 MAC determined using random forest classifier help to reduce the end-to-end latency compared to the default MAC parameters of IEEE 802.15.4 standard. The proposed solution has been published in [5].

Congestion control in IoT low power Networks

Sommaire

5.1 Introduction

IoT low power networks are based on resource-constrained networks such as Wireless Sensor Networks (WSNs) which consist of a collection of small sensor devices with limited resources (energy, bandwidth, processing and memory) and deployed in an area of interest in order to collect the physical conditions of the environment and send them to a sink or a base station. In our daily life, WSN applications include smart agriculture, smart factory, smart homes, and so forth. However, since nodes in a WSN are resource constrained, during the lifetime of a WSN, a congestion can occur at any point in the network when the amount of traffic load exceeds available resources. Congestion in WSNs is generally observed at node or link level, as pointed out in [52] (cf. Figure 5.1). Specifically, where network congestion results in packet losses, an increase of delays of packet transmission and a reduction of the network lifetime [52]. Therefore, it is crucial to design efficient mechanisms for congestion control in resource-constrained networks.

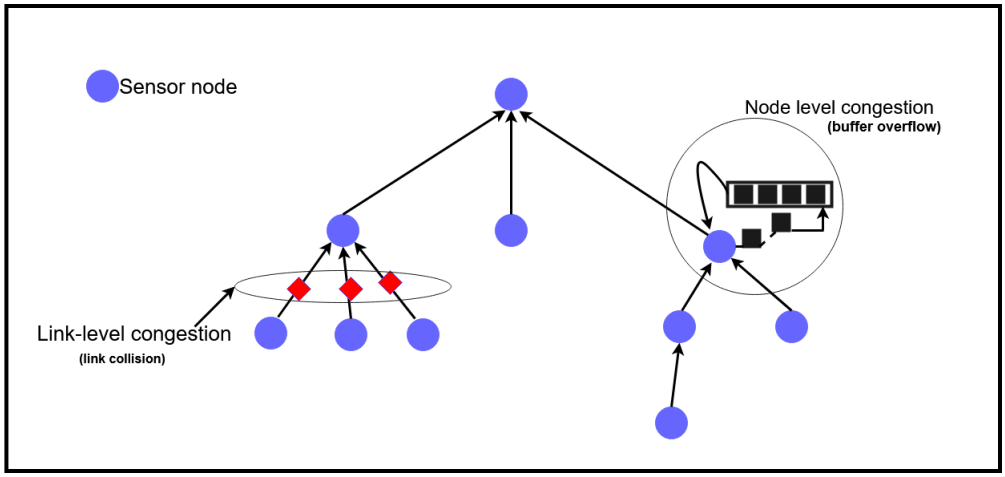


Figure 5.1 – Example of a congested wireless sensor network.

In this chapter, we review the existing solutions for congestion control in IoT low power networks. In this review, we point out the limitations of the existing solutions and motivate the need for a machine learning approach to adress the problem of congestion control in resource-constrained networks. Moreover, we also provide an insight for designing a congestion control solution for resource-constrained networks using imbalanced dataset (a type of dataset where the number of observation per class is unequally distributed).

The remaining of this chapter is organized as follows. In section 5.2 we present the background related to congestion management in IoT low power networks. Then, in section 5.3 we provide an overview of existing solutions for congestion control in IoT low power networks. In section 5.4 we discuss the use of machine learning techniques for congestion control in resource-constrained networks. Finally, section 5.5 concludes this chapter.

5.2 Background

Congestion management mechanisms are generally operating in three steps: congestion detection, congestion notification and congestion control [57].

Step 1: Congestion detection consists in monitoring the resource constrained network in order to identify both the presence and location of a congestion. The identification of congestion is generally done by monitoring parameters such as buffer occupancy and channel load.

Step 2: After the detection of a congestion problem, the upstream or downstream nodes are notified in order to take the right decision to face the congestion.

The upstream or downstream nodes are informed through explicit notification (it consists in sending congestion information in a specific control packet to the upstream or downstream nodes) or implicit notification (it consists in sending congestion information by piggybacking it in a payload of a packet header).

Step 3: After the detection and notification of a congestion problem in the network, a mechanism of control of congestion is performed in order to mitigate the effect of congestion. This can be achieved using techniques such as transmission rate control, load balancing and duty cycle adjustment.

5.3 Existing solutions for congestion control in IoT low power networks

In the literature, a large number of solutions that address the problem of congestion in resource-constrained networks have been proposed. In [157], a solution called CODA (Congestion Detection and Avoidance) has been proposed. This solution uses two strategies to mitigate congestion in resource constrained networks . These strategies are: open-loop hop-by-hop backpressure (this enables an adjustment of the sending rate hop-by-hop) and closed loop multi-source regulation (this helps regulate the sending rate of nodes according to the feedback of the sink). However, the feeback messages used in this proposal may lead to a waste of network resources. To cope with this issue, the authors in [149], proposed an improvement of CODA called Enhanced Congestion Detection and Avoidance (ECODA), which uses dual buffer thresholds and weighted buffer difference for congestion detection. Moreover ECODA uses flexible queue scheduler for packets scheduling and a bottleneck-node-based source sending rate control scheme. On the other hand, a rate-based Fairness-Aware Congestion Control (FACC) protocol has been proposed in [174]. This proposal enables classification of sensor nodes according to their position to the sink in order to achieve an approximately fair bandwidth allocation. The nodes classified as near-source nodes maintain a per-flow state and allocate an approximately fair rate to each passing flow by comparing the incoming rate of each flow and the fair bandwidth share. The nodes classified as near-sink nodes use lightweight probabilistic dropping algorithm based on queue occupancy and hit frequency. Additionally, an upstream hop-by-hop congestion control (UHCC) protocol based on cross-layer design has been proposed in order to mitigate congestion in resource constrained networks [159]. UHCC uses buffer size and packet delivery rate to detect congestion. After congestion detection, the congestion is mitigated through an adjustment of the traffic rate.

However, the above solutions are not able to anticipate the onset of congestion in resource constrained networks since they do not include a predictive mechanism. To cope with this limitation machine learning based solutions for congestion control in resource constrained networks have been proposed. Many of these solutions are based on supervised learning (a category of machine learning algorithms) techniques. Supervised learning techniques allow performing classification or regression using pre-labelled data in order to build a function that allows to determine the class label of unseen inputs.

In [142], the authors proposed a congestion detection model for IoT low power networks. The proposed model operates on the sink node and it is based on a neural network. This neural network model uses the number of active source nodes in the system, the buffer occupancy and the traffic rate to correctly detect the level of congestion. In the same vein, the authors in [101] proposed a model based on classification by regression to detect congestion level in a resource constrained network. Using NS2 network simulator, the authors generated a data set composed of packet loss, packet delivery ratio, packet service time and packet inter-arrival time. This generated data set has been used to provide a model that enables classification of network congestion level. However, these machine learning based approaches are limited to congestion detection. To answer this concern, the authors in [107] proposed a method which includes a congestion control mechanism. The proposed method is based on a neural network, which is embedded in the sink node. This neural network uses a number of parameters such as the delay of data transmission and the node energy to identify the source of congestion and take a decision of reconfiguration of the resource constrained network in order to mitigate the congestion. Furthermore, in [71], a method based on a radial basis function network [1] to improve the performance of routing protocol in a resource constrained network has been proposed. In this proposal, the data loss ratio and memory occupancy have been used to detect congestion in the network at the base station. Then, transmission rate adjustment is performed to mitigate the congestion in resource constrained networks.

In [48], the authors proposed a congestion control scheme for resource-constrained networks based on a neural network. The proposed approach called "Modified Neural Network Wavelet Congestion Control" (MNNWCC) enables congestion detection, notification and control. This solution is implemented at a sink node, which uses a neural network that takes the buffer occupancy status as input to estimate the traffic so that the data transmission rate of nodes can be adjusted

[1] A type of neural artificial neural network that uses radial basis functions as an activation functions

to mitigate the network congestion. However, the proposed machine learning model has not been evaluated in order to assess its accuracy.

On the other hand, in [59], Gholipour et al. proposed a hop-by-hop congestion avoidance method for resource constrained networks. The proposed method is based on the Support Vector Machine (SVM) model and enables performing an adjustement of the transmission rate to mitigate congestion in resource constrained networks. To achieve this, the authors of this proposal generated a dataset composed of the buffer occupancy ratio and the congestion degree to build a model that adjusts the transmission rate that mitigates the congestion in resource constrained networks.The perfomance of their model has been improved by using genetic algorithm to tune the parameters of their SVM model. In the same vein, the works in [85] and [84], improve the classification accuracy of the previous work by using the differential evolution [132] and grey wolf optimization [50] algorithms to tune the SVM parameters.

We provided in Table 5.1, a comparison of the above solutions for congestion management according to the following elements: Congestion detection, Congestion notification, Congestion control, Functioning, Performance metrics, Source of dataset, Imbalanced dataset, Compared with, Remarks. From that comparative study, we observe that most of the existing solutions based on supervised learning techniques for congestion management in resource constrained networks do not consider the presence of imbalanced data during the design of their learning models. In particular, for machine learning based solutions for congestion control, this may lead to biaised classification models, which result in wrong decisions for congestion mitigation and thus cause a reduction of the network performance. To cope with this issue, various data resampling techniques can be used as in [122], [168] and [123], in order to reduce the effect of the skewed class distribution in the learning process .

In the following section, we provide a comparative analysis of different resampling techniques applied for the design of machine learning models for congestion mitigation in resource constrained networks.

Table 5.1 – Comparison of various solutions for congestion management in resource-constrained networks.

Solution	**Congestion detection**	**Congestion notification**	**Congestion control**	**Functioning**	**Performance metrics**	**Source of dataset**	**Imbalanced data**	**Compared with**	**Remarks**
CODA [157]	Queue Length and Channel Load	Explicit	Additive-Increase Multiplicative-Decrease(AIMD)	Distributed	Energy Tax, Fidelity Penalty	-	-	No congestion control, open loop control	Improve the network lifetime
ECODA [149]	Dual buffer thresholds and weighted buffer difference	Implicit	AIMD rate adaptation	Distributed	Throughput, delay, fairness	-	-	CODA	Improve the network lifetime
FACC [174]	Channel busyness and queue length	Explicit	Rate control	Distributed	Packet Loss, Source Rate, Fairness and throughput	-	-	CODA, No Congestion Control	Improve the network lifetime
UHCC [159]	Packet delivery ratio and Buffer size	Implicit	Rate control	Distributed	Throughput, Fairness and Loss Ratio	-	-	PCCP, CCF	Improve the network lifetime
SVM + Genetic Algorithm [59]	Buffer occupancy ratio and Congestion degree	Implicit	Rate adjustment	Distributed	Mean Squared error, packet loss, network, throughput, energy consumed	Generated by the authors	Not considered	Decision tree, Naive bayes classifier, K-nearest neighbor, CODA, ECODA	Improve the network lifetime
SVM + Differential Evolution and Grey Wolf Optimization algorithms [85, 84]	Buffer occupancy ratio and Congestion degree		Rate adjustment	Distributed	Mean Squared error, Mean Absolute error, Root mean square error	Dataset provided by [59]	Not considered	GA-SVM, K-NN, Naive Bayes, and Random Forest	Improve the classification error
Neural networks [107]	Traffic statistic (transmission delay, node energy)	-	Rate adjustment	Centralized	Packet delivery rate, point-to-point delay, Accuracy	Generated by the authors	Not considered	Works in [58, 176]	Improve the network lifetime

Table 5.1 – Comparison of various solutions for congestion management in resource-constrained networks.

Solution	**Congestion detection**	**Congestion notification**	**Congestion control**	**Functioning**	**Performance metrics**	**Source of dataset**	**Imbalanced data**	**Compared with**	**Remarks**
radial basis function of neural networks (RBF) [71]	data loss ratio, memory occupancy	-	Rate adjustement	Centralized	Memory utilization, data loss ration	Generated by the authors	Not considered	Network with and without the proposed controller	Improve the performance of a routing protocol
Neural network [142]	number of participants, buffer occupancy and traffic rate	Explicit	-	Centralized	Mean square error (MSE), Confusion matrix, packet drops	Generated by the authors	Not considered	Network with and without the proposed controller	Provide an accuracte model for congestion detection in resource constrained networks
Neural network [48]	buffer occupancy status	Explicit	Rate adjustment	Centralized	Packets loss ratio, buffer utilization, network energy, packets loss and throughput	Generated by the authors	Not considered	Network with and without the proposed controller	Improve the network QoS
Classification by regression [101]	Packet loss, Packet delivery ratio, packet service time, packet inter-arrival time	-	-	-	Accuracy, Mean Absolute error, root mean square error, true positive rate, false positive rate	Generated by the authors	Not considered	Multilayer perceptron	Accurate model for congestion detection in resource constrained networks

5.4 The use of machine learning for congestion control in IoT low power networks

5.4.1 Motivation

As mentioned earlier, resource constrained networks may experience congestion during their operation when the traffic load exceeds the available resources. This results in resource-constrained networks with poor performance, characterized by a high energy consumption, a long delay for data transmission and so forth. To cope with the congestion problem, a number of machine learning techniques have been proposed. The benefits introduced by machine learning techniques for congestion control in resource-constrained networks can be summarized as follows:

- Machine learning based solutions can help to provide an accurate prediction of the wireless link quality [98]. This is useful for congestion control schemes in resource constrained networks since it enables the selection of the optimal path for data routing with minimal end-to-end delay between a node and the base station.

- Machine learning based solutions in resource constrained networks is beneficial for congestion mitigation in dynamic networks involving an important number of nodes [93]. In fact, in such networks, machine learning can help to predict the onset of congestion according the current state of the network. This allow to take a decision that help to avoid network congestion.

5.4.2 Congestion control with machine learning based on imbalanced data

5.4.2.1 Assumptions and problem formulation

We supposed a wireless sensor network composed of N nodes randomly deployed in an area of interest in order to monitor the environmental parameters such as air quality. After the detection of an event, each sensor node sends its sensed value to a sink or a base station through child-to-parent links in a multihop routing. We assume that sensor devices are homogeneous in terms of wireless communication technology, computational capabilities and have the same amount of initial energy. During the network operation, the problem of congestion can occur due to resource limitations of sensor devices. To enable the design of a supervised learning model for congestion control, a set of parameters are collected.

In resource-constrained networks, some events may occur rarely, causing thus an imbalanced training data. Likewise, due to the limitation in terms of energy of nodes, the collection of the training data may be done only in a given period of time T in order to avoid interfering with the main network services. This may results in having an imbalanced training data.

In this context, the main challenge is to determine for different scenarios, the strategy that helps mitigate congestion in wireless sensor networks using imbalanced data.

5.4.2.2 Description of our experiment

In this subsection, we study of the performance of different resampling techniques used to build machine learning models for congestion mitigation in wireless sensor networks based on imbalanced set of data. For this purpose, we use the dataset provided by the authors in [59]. This dataset is composed of 400 entries. To obtain that dataset, a network deployed in an area of $100m \times 100m$ and composed of 100 nodes has been simulated [59]. During the simulation phase, the buffer occupancy ratio ($\triangle B$), the congestion degree ($\triangle C$), the data transmission rate ($\triangle R$) and the number of packet retransmissions have been collected. $\triangle B$ represents the difference between the buffer occupancy ratio of a given node and the buffer occupancy ratio of its downstream node. This parameter allows an estimation of the state of the buffers. The congestion degree on the other hand allows to evaluate the changing tendency of the queue buffer in a period of time. $\triangle C$ represents the difference of the congestion degree of a node and its downstream node. $\triangle R$ represents the value that enables an adjustement of the transmission rate in order to mitigate the congestion in resource constrained networks. The number of packet retransmissions is determinated based on the values of $\triangle B$, $\triangle C$ and $\triangle R$.

However, the machine learning model proposed in [59] for determining the transmission rate that mitigates the congestion in a resource constrained networks is not efficient since it is based on imbalanced dataset. In fact, as shown in Table 5.2, the number of data points available for the different classes is inequally distributed. In such situations, the proposed machine learning model cannot be used in a given application as shown in [7]. To tackle this issue, we propose in this work to generate synthetic data to build a machine learning model that can enable an adjustement of the transmission rate in order to mitigate the problem of congestion in resource constrained networks.

Figure 5.2, 5.3 and 5.4 respectively depict the scatter plot of the original dataset, the scatter of artificial data obtained by performing over-sampling and the scatter

Table 5.2 – Analysis of the dataset proposed in [59].

Class	Number of data points
Class 1	43/400
Class 2	44/400
Class 3	67/400
Class 4	101/400
Class 5	79/400
Class 6	30/400
Class 7	35/400
Class 8	1/400

plot of the artificial dataset obtained by performing under-sampling. This illustrates the difference of data distribution introduced by each resampling method.

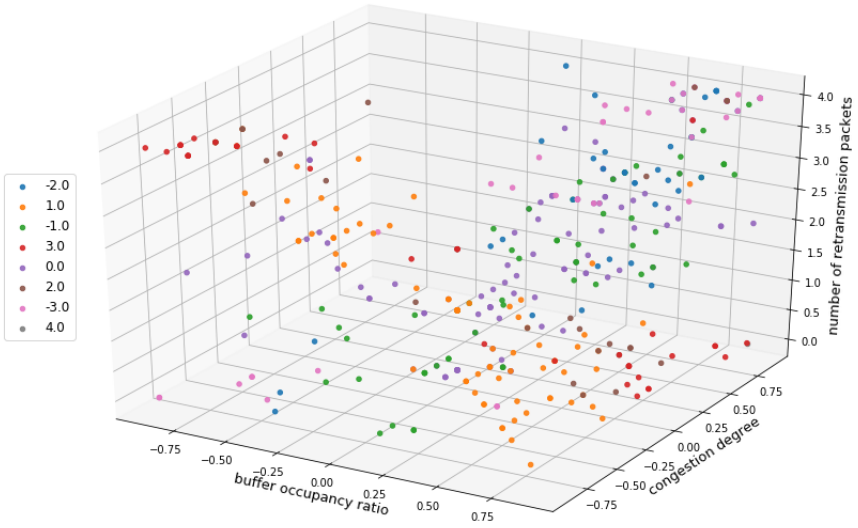


Figure 5.2 – Scatter plot of the original imbalanced dataset.

In order to mitigate the problem of congestion in the network, we have to adapt the transmission rate of a given node according to the buffer occupancy ratio and the congestion degree. We designed a solution based on a machine learning model trained using imbalanced dataset and it is divided into these steps:

- The first step is to build the machine learning model that enables to determine the transmission rate that mitigates the problem of congestion in resource constrained networks. This is done following the different steps depicted in Figure 5.5. The imbalanced dataset is used to generate an artificial dataset that we used to build our machine learning model. This step is followed by a validation of our machine learning model according to some metrics.

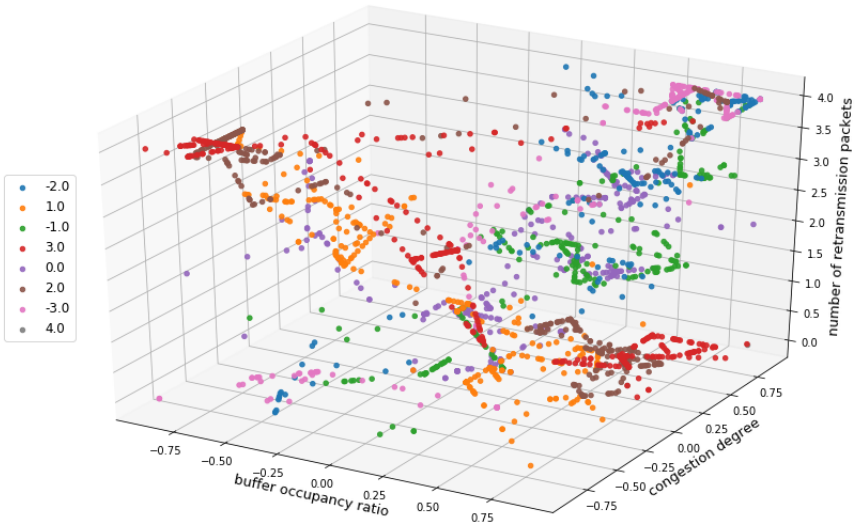


Figure 5.3 – Scatter Plot of the articial data generated using BorderlineSMOTE.

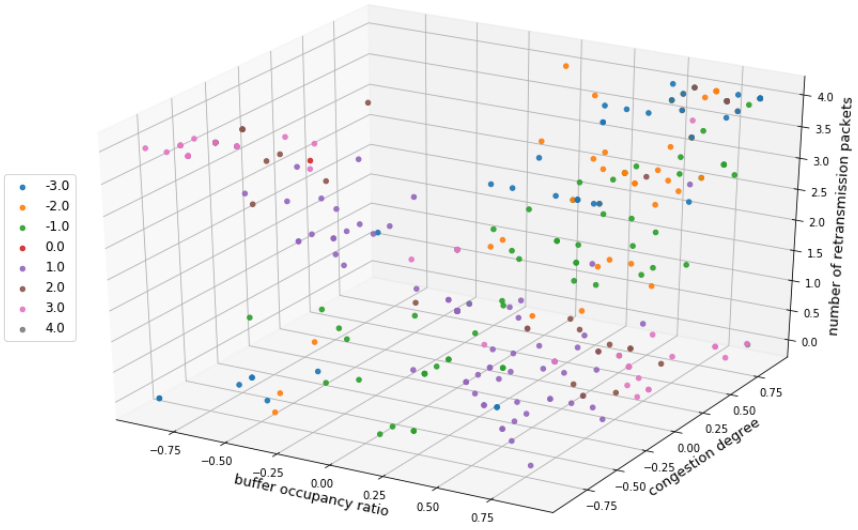


Figure 5.4 – Scatter Plot of the articial data generated using RandomUnderSampler.

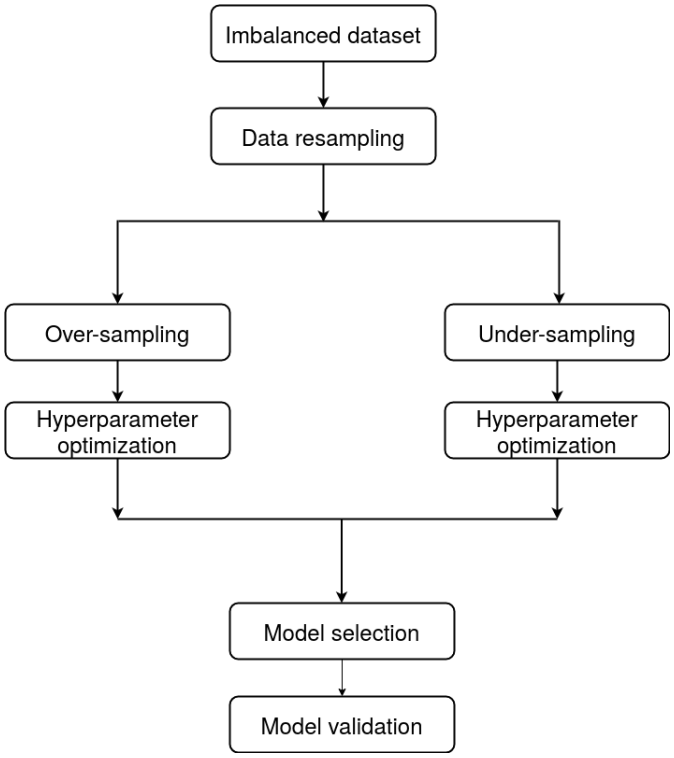


Figure 5.5 – Worflow of our proposal for congestion control.

- The second step concerns the functioning of our solution to update the transmission rate of sensor nodes. This task is perfomed periodically by each parent node after receiving the congestion degree and the buffer occupancy from one of its child nodes.

We chose to evaluate, the most used machine learning algorithms for congestion control in resource constrained networks. In particular, we considered Support Vector Machine (SVM), Random Forest, K-Nearest Neighbor (KNN), Decision Tree and Naives bayes algorithms.

5.4.2.3 Performance evaluation

We implemented our machine learning models using sklearn [119] and imbalanced-learn library [95]. Sklearn is a library which offers implementation of various machine learning algorithms, while imbalanced-learn is a library which provides various techniques for data re-sampling. The evaluation has been done based on the following metrics: G-mean, Index of Balanced Accuracy(IBA), Accuracy, Precision and F-measure.

We compared different resampling strategies in context of congestion control in resource constrained networks based on imbalanced data. In fact, we performed over-sampling and under-sampling in order to balance the dataset provided in [59]. Then,

we used the generated artificial dataset to build an efficient machine models that helps to determine the transmission rate that mitigates the congestion in a resource constrained network. Figure 5.6, Figure 5.7 and Figure 5.8 show respectively :

- The output of the classification_report function for SVM model based on imbalanced data;
- The output of the classification_report function for SVM model based on data generated using under-sampling technique;
- The output of the classification_report function for SVM model based on the balanced data generated using over-sampling technique.

We can see that oversampling technique works better compared to under-sampling since it improves the precision of classes that are not handled by the machine learning model based on imbalanced data or artificial data generated using under-sampling technique.

	precision	recall	f1-score	support
Class 1	0.45	0.29	0.36	17
Class 2	0.00	0.00	0.00	9
Class 3	0.00	0.00	0.00	10
Class 4	0.40	0.78	0.53	18
Class 5	0.64	0.64	0.64	11
Class 6	0.00	0.00	0.00	5
Class 7	0.67	0.60	0.63	10
accuracy			0.40	80
macro avg	0.31	0.33	0.31	80
weighted avg	0.36	0.40	0.36	80

Figure 5.6 – Output of the classification_report function for imbalanced data.

	precision	recall	f1-score	support
Class 1	0.40	0.50	0.44	8
Class 2	0.29	0.20	0.24	10
Class 3	0.25	0.15	0.19	13
Class 4	0.00	0.00	0.00	1
Class 5	0.40	0.86	0.55	14
Class 6	0.00	0.00	0.00	5
Class 7	0.50	0.30	0.37	10
accuracy			0.38	61
macro avg	0.26	0.29	0.26	61
weighted avg	0.33	0.38	0.32	61

Figure 5.7 – Output of the classification_report function for balanced data using under-sampling.

	precision	recall	f1-score	support
Class 1	0.32	0.30	0.31	50
Class 2	0.11	0.13	0.12	38
Class 3	0.22	0.36	0.27	53
Class 4	0.50	0.02	0.03	61
Class 5	0.35	0.72	0.47	75
Class 6	0.49	0.60	0.54	83
Class 7	0.50	0.01	0.02	81
accuracy			0.33	441
macro avg	0.35	0.31	0.25	441
weighted avg	0.38	0.33	0.27	441

Figure 5.8 – Output of the classification_report function for balanced data using over-sampling.

We built various machine learning models using the artificial data we generated and compared them according to the metrics presented in section 4.4.2. We used 80% of that dataset for training and the remaining for the test. We summarized in Table 5.3 the results of the comparison of random forest, SVM, KNN and Naive Bayes. We can see that, machine learning models based on data generated using over-sampling provide good performance compared to machine learning models built using data generated with under-sampling technique.

Table 5.3 – Performance comparison.

Metric	Resampling	Random forest	SVM	KNN	Decision tree	Naive Bayes
F-measure	over-sampling	**0.5535**	0.26782	0.5391	0.5376	0.2337
	under-sampling	**0.5126**	0.2230	0.4449	0.2034	0.1184
Geometric mean	over-sampling	0.4897	0.1433	**0.5145**	0.49028	0.0
	under-sampling	**0.4934**	0.0	0.4451	0.0	0.0
Index balanced accuracy	over-sampling	0.2302	0.0193	**0.2541**	0.2305	0.0
	under-sampling	**0.2341**	0.0193	0.1898	0.0	0.0
Precision	over-sampling	**0.5560**	0.3835	0.5558	0.5396	0.2374
	under-sampling	**0.5681**	0.2128	0.5309	0.2447	0.1144
Accuracy	over-sampling	**0.5578**	0.3287	0.5328	0.5396	0.2857
	under-sampling	**0.4918**	0.2950	0.4262	0.2131	0.1311

5.4.2.4 Discussion

In resource-constrained networks, having an imbalanced data may occur frequently. Machine learning models developed using imbalanced data generally exhibit a worst predictive accuracy in the minority class. In particular this may pose problem for congestion congestion control solutions based on machine learning models build using imbalanced data. To tackle this issue, it may be necessary to perform a resampling of the imbalanced data so that machine learning model can perform well and enable an optimization of the network performance through an appropriate decision of congestion mitigation. As an example, in this work, we showed that oversampling technique may help improving the accuracy performance of machine learning models based on small dataset. We provide in Table 5.4 a guide for choosing a resampling technique according to the network architecture.

Table 5.4 – A guide for applying a resampling technique.

Resampling method	Architecture of the network	Training dataset size	Comment
Over-sampling	Distributed	Small	This strategy may help to balance the dataset and thus improve the accuracy of the machine learning model.
		Large	This strategy may not be adapted for balancing a large dataset [62]. Moreover, manipulating a large dataset on resource constrained nodes may lead to a rapid exhaustion of resources such as memory and battery
	Centralized	Small	This strategy may help to balance the dataset and improve the accuracy of the machine learning model.
		Large	This strategy may not be adapted for balancing a large dataset [62].
Under-sampling	Distributed	Small	This strategy may not be adapted for balancing a small dataset [62].
		Large	This strategy may help to balance the dataset and thus improve the accuracy of the machine learning model. However, due to resource constraints, it may be necessary to perform the training of the machine learning model on a device with powerful resources.
	Centralized	Small	This strategy may not be adapted for balancing a small dataset
		Large	This strategy may help to balance the dataset and thus enable an improvement of the machine learning accuracy.

5.5 Conclusion

In this chapter, we provided an overview of existing solutions for congestion management in resource-constrained networks and pointed out their limitations. We also motivate the need for using machine learning techniques to adress the problem of congestion control in IoT low power networks. For machine learning based approaches, we showed that it is necessary to pay attention to the distribution of the training data so as to avoid having a machine learning model that leads to wrong decisions for congestion mitigation. Furthermore, we showed that for a small training dataset size, over-sampling technique may be useful for balancing the training data in both distributed and centralized IoT low power networks. Additionally, we provided a guide that may help to choose a strategy for resampling training data in context of IoT network low power networks.

CIB: Congestion Information Block for IoT low power networks

Sommaire

6.1 Introduction

IoT low power networks can be defined as networks composed of embedded devices with limited power, memory, and processing resources, and typically communicating over a wireless links. Today, such networks are used in various applications of our daily life such as smart factory, smart home, smart transportation, smart healthcare, smart agriculture and so forth [65]. However, due to their limited resources, IoT low power networks will face network congestion whenever the traffic load exceeds the available capacity at any point in the network [57]. This situation contributes in decreasing the network performance. This is typically characterized by the increase of packet loss ratio and the degradation of the throughput of the wireless channel.

In order to cope with the congestion problem in IoT low power networks, various congestion control mechanisms have been proposed [26]. These mechanisms operate in three steps, including congestion detection, congestion notification and congestion mitigation. Congestion detection refers to the process of monitoring IoT low power networks in order to find both the presence and location of a congestion. This is done by monitoring parameters such as buffer occupancy (queue length), and channel load. Congestion notification allows the node concerned by the congestion

problem or risk to notify the upstream or downstream nodes so that the appropriate congestion control decision can be taken. In general, the notification of the upstream or downstream nodes is done through Explicit Congestion Notification (ECN) or an Implicit Congestion Notification (ICN). An ECN notification consists in sending congestion information in a specific control packet to the upstream or downstream nodes [171]. An ICN notification consists in sending congestion information by piggybacking the congestion information in a payload of a packet header [26]. After the detection and notification of a congestion problem in the network, the mitigation strategy of the network congestion is done using various techniques such as transmission rate control, load balancing and duty cycle adjustment.

Congestion notification represents an important step for congestion control in IoT low power networks since it allows taking the right decision against congestion, according to the network state. However, in an event driven application of IoT low power networks (e.g. forest-fire detection), existing congestion notification schemes like ECN may generate an important network traffic that will increase the risk of network congestion, and thus cause the deterioration of the network performance.

In this chapter, we propose a novel congestion notification scheme that enables the transmission of congestion state of IoT low power devices in a given routing path into a Congestion Information Block (CIB) sent to a central entity (the network manager). The proposed scheme allows an efficient aggregation of congestion state of nodes in a given routing path into the CIB by using binary values. Compared to an ECN scheme, our proposed solution offers a good performance in terms of network throughput, network overhead and it offers low divergence (regarding the time of observation) between congestion observed by the network manager and the real congestion of nodes.

The rest of the chapter is organized as follows: section 6.2 describes our solution; this is followed by the performance evaluation of our proposal in section 6.3; and the conclusion of this chapter is presented in section 6.4.

6.2 Proposed scheme for congestion state notification

6.2.1 Assumption and problem formulation

We consider an event-driven IoT low power network composed of k nodes scattered in a 2-dimentional area. The set of nodes is represented by $S = \{n_1, n_2, \cdots, n_k\}$. Each node has a unique identifier $i \in \{1, \cdots, k\}$. We assume

that all nodes are homogeneous (in terms of wireless communication technology and computational capabilities) and are topologically static. We also assume that our network is managed by a central entity, which is responsible for mitigating network congestion (cf. Figure 6.1). Moreover, we supposed that each node is monitoring its buffer occupancy in order to notify the network manager. A threshold value θ is used by each node so that when buffer occupancy ratio is above θ, the node is considered as congested otherwise the node is not congested.

The problem consists in providing an optimal method to notify the congestion state of nodes to the network manager while ensuring good network performance.

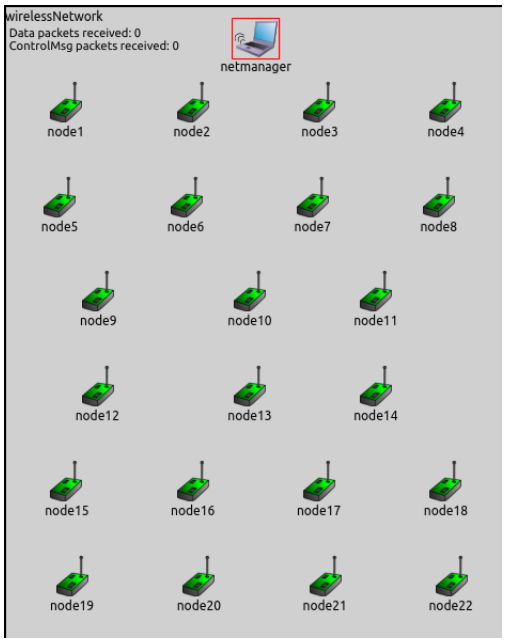


Figure 6.1 – Example of an IoT low power network on Omnet++ simulator.

6.2.2 Design of the proposed congestion notification scheme

In order to solve the above mentioned problem, we propose a novel scheme for congestion notification that allows to efficiently aggregate the congestion state of nodes in a given routing path into a block called Congestion Information Block (CIB). Compared to ECN and ICN, the proposed scheme provides the following benefits:

- No additional packet is required to send congestion state of nodes to the network manager.
- Only a single packet is require to send the congestion state of nodes in a given routing path.

Figure 6.2 illustrates the transmission of the CIB toward the network manager. The transmission of the CIB is initiated by leaf nodes during the transmission of a data packet. For each packet to be sent to network manager, each node located in the routing path will insert its congestion state. The CIB is inserted into the payload of a packet as depicted in Figure 6.3. Typically, each node inserts a binary value representing its Congestion Information (CI) in the packet payload according to this rule:

$$CI = \begin{cases} 1 & if\ bufferOccupancy \geqslant \theta \\ 0 & if\ bufferOccupancy < \theta \end{cases} \tag{6.1}$$

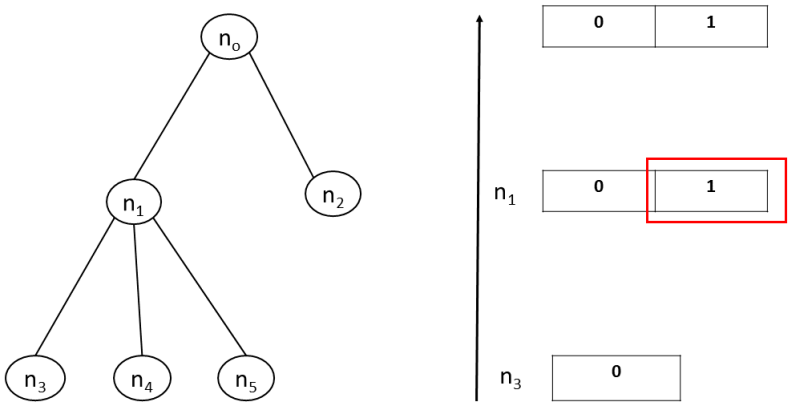


Figure 6.2 – Transmission of a CIB of a given routing path.

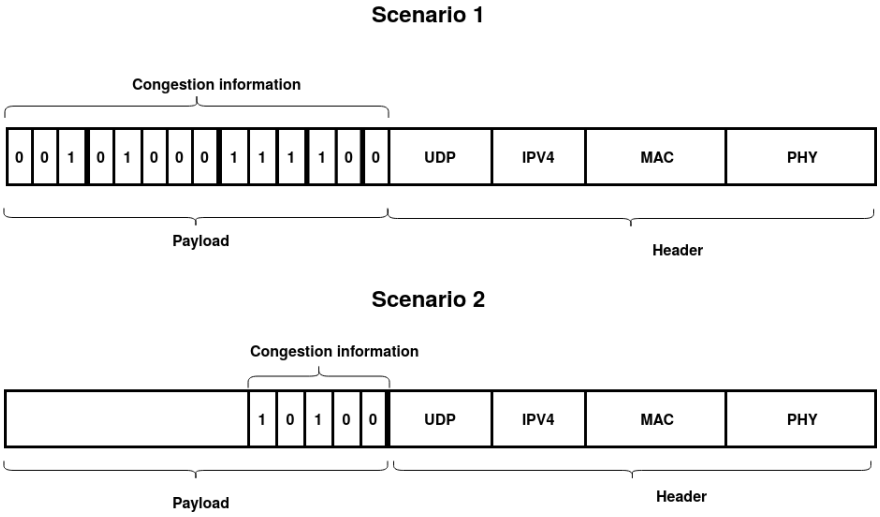


Figure 6.3 – Example of a CIB block in a data packet (case of a UDP packet).

We provide in Figure 6.4 an overview of different operations related to CIB generation by leaf nodes. In Figure 6.5, we give an overview on operations executed by intermediate nodes after the reception of a packet with a CIB.

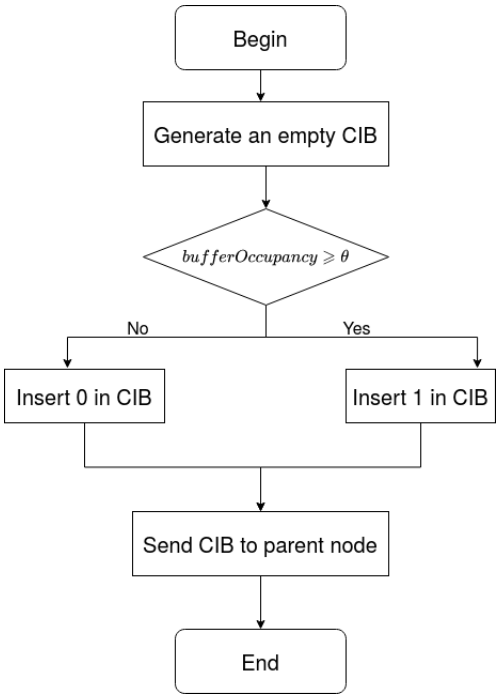


Figure 6.4 – Generation of a CIB by a leaf node.

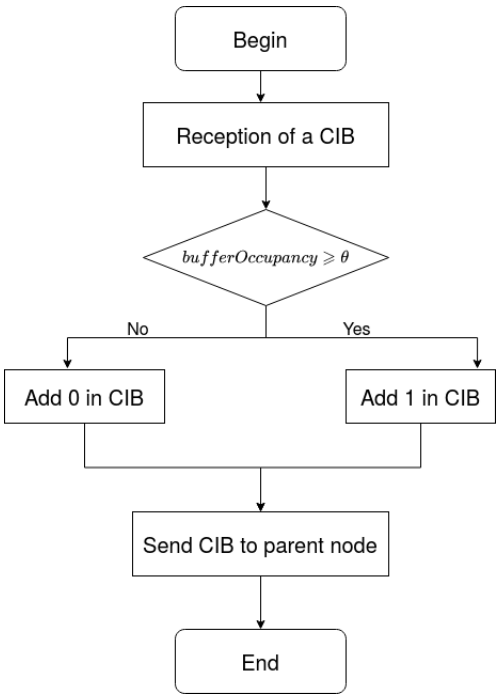


Figure 6.5 – Insertion of a CI into a CIB by an intermediate node.

To process the CIB, the network manager verifies each binary value along with

its index (node identifier) in the CIB so that all nodes concerned by the congestion control decision are known by the manager (cf. Figure 6.6). The congestion control decision may consist, for instance in a reconfiguration of the routing table or an adjustment of the sending rate of nodes.

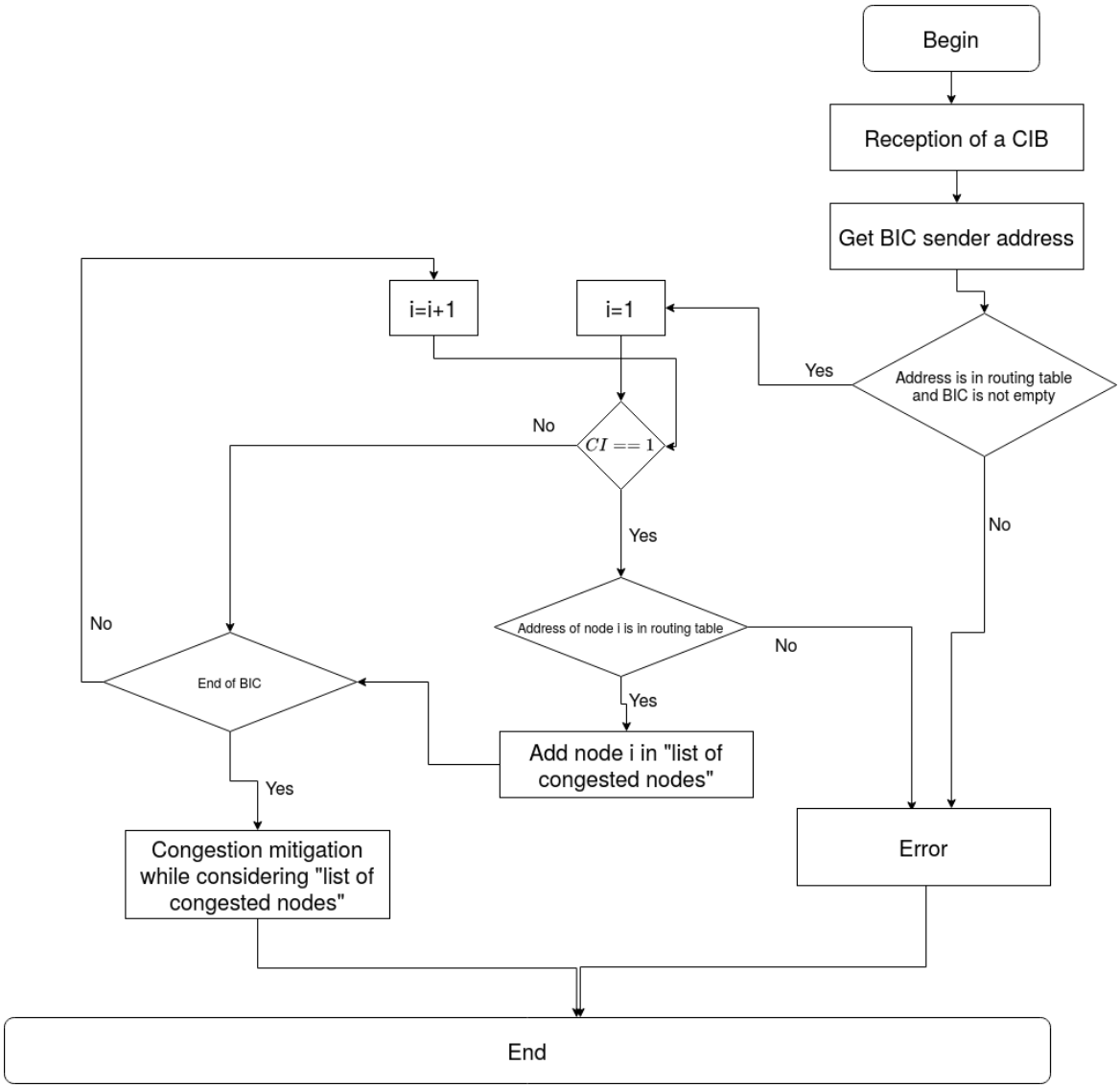


Figure 6.6 – Analyze of the CIB by the network manager.

6.3 Simulation and performance evaluation

6.3.1 Simulation environment

We evaluate the performance of our proposed scheme for congestion notification by performing series of simulation in various scenarios using Omnet++, an object-oriented modular discrete event simulator [154]. Moreover, we simulated the realistic

behavior of nodes using INET framework, an open-source OMNeT++ model suite for wired, wireless and mobile networks [1]. The main parameters used in the simulation are shown in Table 6.1.

Table 6.1 – Simulation parameters.

Parameter	**Value**
Bitrate	100kps
Max queue size	50 packets
Number of Nodes	23
Number of ECN sources	22
Number of CIB sources	9
Data packet generation	exponential(8ms)
CIB/ECN generation	exponential(15ms), exponential(25ms)
Experiment duration	120s
Communication range	200m
Deployment area	1000m x 1000m

To evaluate the performance of our proposed scheme for congestion notification, we performed a comparison with ECN [171], a well known scheme for congestion notification.

6.3.2 Results and discussion

1) *Control messages overhead:* In Figure 6.7, 6.8, 6.9 and 6.10 we plotted the percentage of data packets and notification messages received by the network when varying the sending interval of CIB and ECN. Overall, our proposed scheme for congestion notification with CIB generates less control messages compared to ECN.

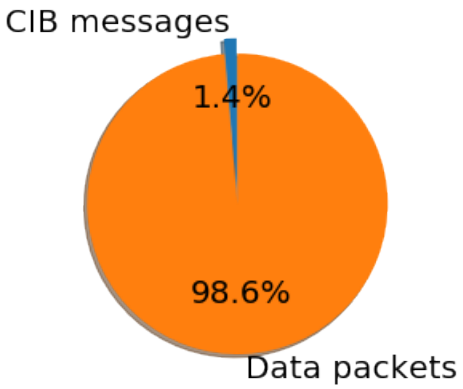


Figure 6.7 – Traffic comparison (CIB sending Interval = exponential (15ms))

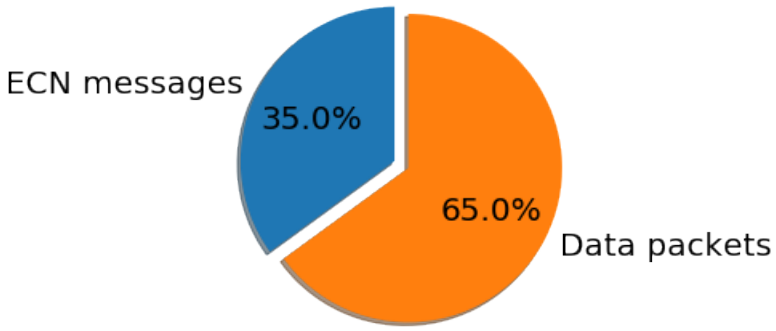


Figure 6.8 – Traffic comparison (ECN sending Interval = exponential (15ms))

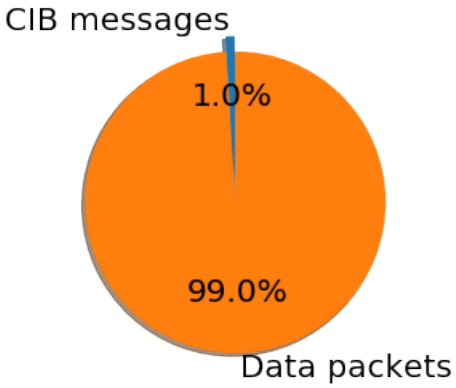


Figure 6.9 – Traffic comparison (CIB sending Interval = exponential (25ms))

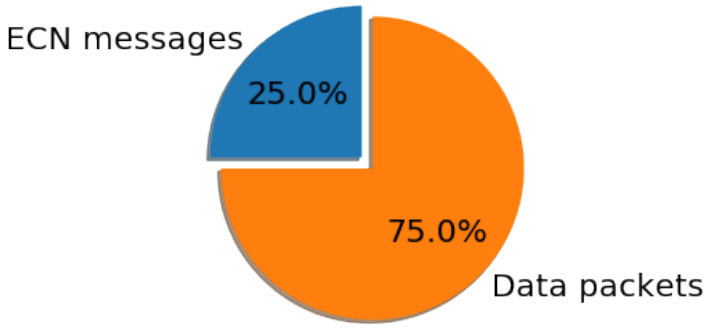


Figure 6.10 – Traffic comparison (ECN sending Interval = exponential (25ms))

2) *Throughput:* Figure 6.11 and 6.12 show different throughput curves obtained when simulating CIB and ECN notification schemes. The throughput of CIB is less than that of ECN.

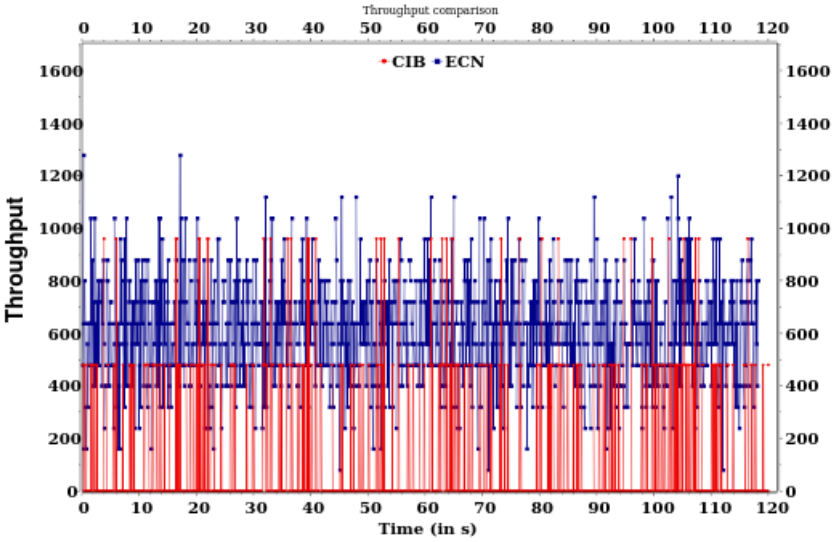


Figure 6.11 – Troughput comparison (sending Interval of control messages = exponential 15ms)

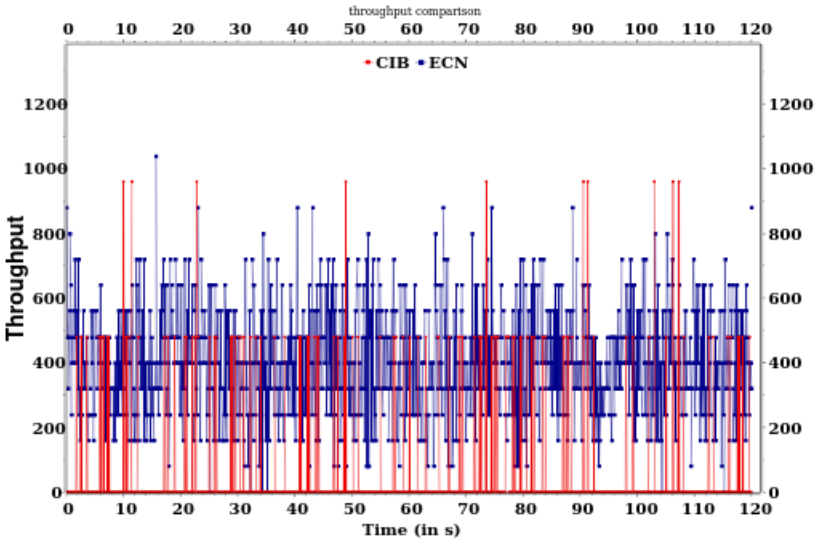


Figure 6.12 – Troughput comparison (sending Interval of control messages = exponential 25ms)

3) *Divergence between congestion observed by network manager and the real congestion state at nodes:* We measured the mean time necessary to notify the network manager that the state of node(s) changed. We observed that our proposed scheme for congestion notification is faster than ECN (cf. Figure 6.13).

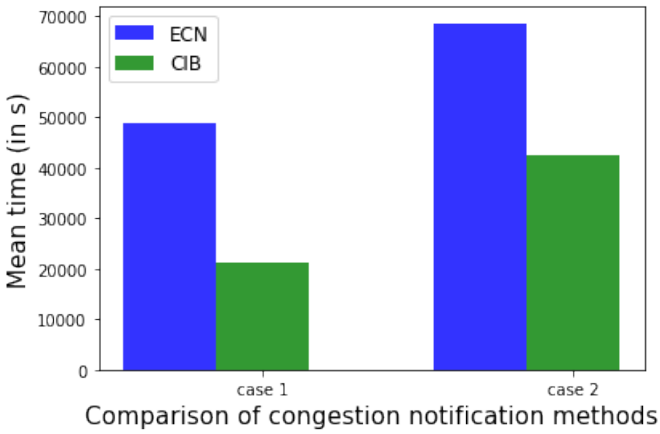


Figure 6.13 – Divergence between observed and real congestion states.

6.4 Conclusion

In this chapter, we proposed a novel scheme for congestion notification in IoT low power networks. The proposed scheme allows to efficiently aggregate the congestion state of nodes in a given routing path into a single data packet by using a CIB, which contains binary values representing the congestion state of nodes. The simulations results show that our proposed scheme performs well compared to ECN in terms of network throughput, control messages overhead and it offers low divergence between the congestion observed by the network manager and the real congestion of nodes.

Conclusion and Future Research Directions

Sommaire

7.1 Conclusion

IoT is a new paradigm that allows the interconnection of various things such as industrial sensors, actuators, smart home appliances and RFID tags in order to facilitate and/or enrich experience in various domains such as transportation, agriculture, healthcare and so forth. The research on IoT has attracted the attention of researchers from academic and industry sectors due to the potential benefits of IoT applications in our daily lives. Nowadays, IoT landscape is characterized by the presence of billions of heterogeneous devices with limited storage capacity, processing capabilities, and energy, that are communicating over error prone wireless links. Due to that limitation, IoT networks are facing many problems of performance, including link quality deterioration, network congestion and failure of devices. In this context, an efficient management of IoT low power networks becomes important in order to improve the network performance. Nonetheless, achieving efficient management of IoT networks is not straightforward because of IoT characteristics and constraints (e.g. devices resource constraints) that raise a number of challenges (e.g. scalability, energy efficiency and fault tolerance).

In this thesis we addressed the problem of management of IoT low power networks and proposed novel solutions for efficient management of those resource-constrained networks.

First, we provided a background knowledge on IoT low power networks and a comparative analysis of existing solutions for management of IoT low power networks

based on different requirements. Based on the limitation of existing solutions for management of IoT low power networks, we proposed in chapter 3, intelligent solutions for determining the efficient transmission power of RPL networks. Our proposed solutions are based on a deep neural network model, which uses nodes position to estimate the efficient transmission power of RPL networks. We performed extensive simulations of proposed solutions on COOJA network emulator in order to demonstrate its efficiency. The results of performance evaluation show that our proposed solutions can help to define the transmission power that enables a reduction of the network energy consumption while maintaining the network connectivity.

On the other hand, we proposed in chapter 4, a novel solution to efficiently configure the parameters of IEEE 802.15.4 MAC in dynamic IoT low power networks. The proposed solution is based on a supervised learning technique and it enables to select the IEEE 802.15.4 MAC parameters according to the network and data traffic characteristics. The results of performance evaluation of our proposal show that the MAC parameters defined using our solution help to reduce the end-to-end delay for data transmission compared to the default MAC parameters of IEEE 802.15.4 standard.

Further, in chapter 5, we proposed a comparative analysis of solutions for congestion control in IoT low power networks and pointed out their limitation and the benefits for having machine learning based solution for congestion control in such resource-constrained networks. Moreover, we also provided a guide for designing a congestion control solution for resource-constrained networks using imbalanced data.

Furthermore, in chapter 6, we proposed a novel congestion notification scheme that enables efficient aggregation of congestion state of nodes in a given routing path into a Congestion Information Block, which contains binary values representing the congestion states of nodes. The obtained results of performance evaluation show that our proposal perform well in terms of network throughput, network overhead and offers low divergence between congestion observed by the network manager and the real congestion of nodes compared to the well known ECN.

7.2 Future Research Directions

Our solutions for management of IoT low power networks presented in this thesis have led to encouraging results. For the next steps, we plan to address the following issues:

1) As future work, we envisage to extend our solution that allows to determine the efficient transmission power of RPL networks in order to meet the scalability

requirement of IoT low power networks. This goal could be achieved by including clustering techniques in our proposal.

2) We plan to further analyze our solution that enables to define the IEEE 802.15.4 MAC parameters to evaluate its energy efficiency. Moreover, we intend to implement our solution on real IoT devices in order to perform an evaluation of its performance in real physical environment.

3) We plan to investigate an online learning based solution for congestion control in resource-constrained networks by considering the insight we provided in chapter 5. In fact, as shown in [163], in various domains, the information required for learning is rarely available a priori. In such cases, an online learning can be used to build machine learning models. However, since IoT lower networks are suffering from resources limitation, the proposed solution have to be energy efficient and it should not introduce an overhead in the networks.

4) In our future work, we plan to extend our proposed scheme for congestion notification that enables efficient aggregation of congestion state of nodes in a given routing path into a Congestion Information Block, by integrating a congestion mitigation scheme.

Bibliography

[1] *INET Framework*, 2020 (accessed September 09, 2020). `https://inet.omnetpp.org/`.

[2] Kenshin Abe, Issei Sato, and Masashi Sugiyama. Solving np-hard problems on graphs by reinforcement learning without domain knowledge. *Simulation*, 1:1–1, 2019.

[3] Moussa Aboubakar, Mounir Kellil, Abdelmadjid Bouabdallah, and Pierre Roux. Toward intelligent reconfiguration of rpl networks using supervised learning. In *2019 Wireless Days (WD)*, pages 1–4. IEEE, 2019.

[4] Moussa Aboubakar, Mounir Kellil, Abdelmadjid Bouabdallah, and Pierre Roux. Using machine learning to estimate the optimal transmission range for rpl networks. In *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–5. IEEE, 2020.

[5] Moussa Aboubakar, Pierre Roux, Mounir Kellil, and Abdelmadjid Bouabdallah. An efficient and adaptive configuration of IEEE 802.15.4 MAC for communication delay optimisation. In *2020 11th International Conference on Network of the Future (NoF) (NoF 2020)*, University of Bordeaux, France, October 2020.

[6] François Aïssaoui, Samuel Berlemont, Marc Douet, and Emna Mezghani. A semantic model toward smart iot device management. In *Workshops of the International Conference on Advanced Information Networking and Applications*, pages 640–650. Springer, 2020.

[7] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In *European conference on machine learning*, pages 39–50. Springer, 2004.

[8] Bilal R Al-Kaseem, Hamed S Al-Raweshidy, Yousif Al-Dunainawi, and Konstantinos Banitsas. A new intelligent approach for optimizing 6lowpan mac layer parameters. *IEEE Access*, 5:16229–16240, 2017.

[9] Hayder AA Al-Kashoash and Andrew H Kemp. Comparison of 6lowpan and lpwan for the internet of things. *Australian Journal of Electrical and Electronics Engineering*, 13(4):268–274, 2016.

[10] M Udin Harun Al Rasyid, Isbat Uzzin Nadhori, Amang Sudarsono, and Ridho Luberski. Analysis of slotted and unslotted csma/ca wireless sensor network for e-healthcare system. In *2014 International Conference on Computer, Control, Informatics and Its Applications (IC3INA)*, pages 53–57. IEEE, 2014.

[11] Atif Alamri, Wasai Shadab Ansari, Mohammad Mehedi Hassan, M Shamim Hossain, Abdulhameed Alelaiwi, and M Anwar Hossain. A survey on sensor-cloud: architecture, applications, and approaches. *International Journal of Distributed Sensor Networks*, 9(2):917923, 2013.

[12] Vito Albino, Umberto Berardi, and Rosa Maria Dangelico. Smart cities: Definitions, dimensions, performance, and initiatives. *Journal of urban technology*, 22(1):3–21, 2015.

[13] Open Mobile Alliance. Oma device management representation protocol. *Approved Version*, 1(1), 2010.

[14] Mohammad Abu Alsheikh, Shaowei Lin, Dusit Niyato, and Hwee-Pink Tan. Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *IEEE Communications Surveys & Tutorials*, 16(4):1996–2018, 2014.

[15] Arkessa. *Arkessa Services*, 2018 (accessed March 20, 2018). `http://www.arkessa.com/iot-m2m-connectivity-services/`.

[16] Arm. *Mbed IoT Platform*, 2018 (accessed January 25, 2018). `https://www.mbed.com/en/platform/`.

[17] Kevin Ashton et al. That 'internet of things' thing. *RFID journal*, 22(7):97–114, 2009.

[18] Zahra Aslani and Hadi Sargolzaey. Improving the performance of rpl routing protocol for internet of things. *Journal of Computer & Robotics*, 10(2):69–75, 2017.

[19] Autodesk. *Autodesk Fusion Connect*, 2018, (accessed March 12, 2018). `https://autodeskfusionconnect.com/`.

[20] Julio Barbancho, Carlos Leon, Javier Molina, and Antonio Barbancho. Giving neurons to sensors. qos management in wireless sensors networks. In *2006 IEEE Conference on Emerging Technologies and Factory Automation*, pages 594–597. IEEE, 2006.

[21] Samaresh Bera, Sudip Misra, Sanku Kumar Roy, and Mohammad S Obaidat. Soft-wsn: Software-defined wsn management system for iot applications. *IEEE Systems Journal*, 12(3):2074–2081, 2016.

[22] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems*, pages 2546–2554, 2011.

[23] Purvi Bhimani and Gaurang Panchal. Message delivery guarantee and status update of clients based on iot-amqp. In *Intelligent Communication and Computational Technologies*, pages 15–22. Springer, 2018.

[24] Noor Zuriatunadhirah binti Zubir, Aizat Faiz Ramli, and Hafiz Basarudin. Optimization of wireless sensor networks mac protocols using machine learning; a survey. In *2017 International Conference on Engineering Technology and Technopreneurship (ICE2T)*, pages 1–5. IEEE, 2017.

[25] Martin Bjorklund et al. Yang-a data modeling language for the network configuration protocol (netconf). 2010.

[26] Atousa Bohloulzadeh and Mehri Rajaei. A survey on congestion control protocols in wireless sensor networks. *International Journal of Wireless Information Networks*, pages 1–20, 2020.

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

[28] Joel W Branch, Chris Giannella, Boleslaw Szymanski, Ran Wolff, and Hillol Kargupta. In-network outlier detection in wireless sensor networks. *Knowledge and information systems*, 34(1):23–54, 2013.

[29] Gary Brassington. Mean absolute error and root mean square error: which is the better metric for assessing model performance? In *EGU General Assembly Conference Abstracts*, volume 19, page 3574, 2017.

[30] Simone Brienza, Domenico De Guglielmo, Giuseppe Anastasi, Marco Conti, and Vincenzo Neri. Strategies for optimal mac parameter setting in ieee 802.15. 4 wireless sensor networks: A performance comparison. In *2013 IEEE Symposium on Computers and Communications (ISCC)*, pages 000898–000903. IEEE, 2013.

[31] Carriots. *Carriots - Internet of Things Platform*, 2018 (accessed january 04, 2018). `https://carriots.com/`.

[32] Yuchao Chang, Xiaobing Yuan, Baoqing Li, Dusit Niyato, and Naofal Al-Dhahir. A joint unsupervised learning and genetic algorithm approach for topology control in energy-efficient ultra-dense wireless sensor networks. *IEEE Communications Letters*, 22(11):2370–2373, 2018.

[33] Haksoo Choi, Nakyoung Kim, and Hojung Cha. 6lowpan-snmp: Simple network management protocol for 6lowpan. In *2009 11th IEEE International Conference on High Performance Computing and Communications*, pages 305–313. IEEE, 2009.

[34] François Chollet et al. Keras: The python deep learning library. *Astrophysics Source Code Library*, 2018.

[35] Samira Chouikhi, Inès El Korbi, Yacine Ghamri-Doudane, and Leila Azouz Saidane. A survey on fault tolerance in small and large scale wireless sensor networks. *Computer Communications*, 69:22–37, 2015.

[36] Mario Collotta, Arcangelo Lo Cascio, Giovanni Pau, and Gianfranco Scatá. A fuzzy controller to improve csma/ca performance in ieee 802.15. 4 industrial wireless sensor networks. In *2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–4. IEEE, 2013.

[37] Salvatore Costanzo, Laura Galluccio, Giacomo Morabito, and Sergio Palazzo. Software defined wireless networks: Unbridling sdns. In *2012 European Workshop on Software Defined Networking*, pages 1–6. IEEE, 2012.

[38] Kalyan Das, Satyabrata Das, Rabi K Darji, and Ananya Mishra. Energy efficient model for the sensor cloud systems. In *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, pages 373–375. IEEE, 2017.

[39] Soumya Kanti Datta, Amelie Gyrard, Christian Bonnet, and Karima Boudaoud. onem2m architecture based user centric iot application development. In *2015 3rd International Conference on Future Internet of Things and Cloud*, pages 100–107. IEEE, 2015.

[40] Alejandro De Gante, Mohamed Aslan, and Ashraf Matrawy. Smart wireless sensor network management based on software-defined networking. In *2014 27th Biennial Symposium on Communications (QBSC)*, pages 71–75. IEEE, 2014.

[41] Bruno Trevizan De Oliveira, Lucas Batista Gabriel, and Cintia Borges Margi. Tinysdn: Enabling multiple controllers for software-defined wireless sensor networks. *IEEE Latin America Transactions*, 13(11):3690–3696, 2015.

[42] Rodolfo de Paz Alberola and Dirk Pesch. Duty cycle learning algorithm (dcla) for ieee 802.15. 4 beacon-enabled wireless sensor networks. *Ad Hoc Networks*, 10(4):664–679, 2012.

[43] Henock Mamo Deberneh and Intaek Kim. Predicting output power for nearshore wave energy harvesting. *Applied Sciences*, 8(4):566, 2018.

[44] Jing Deng, Yunghsiang S Han, Po-Ning Chen, and Pramod K Varshney. Optimal transmission range for wireless ad hoc networks based on energy efficiency. *IEEE Transactions on Communications*, 55(9):1772–1782, 2007.

[45] Mahidhar Dwarampudi and NV Reddy. Effects of padding on lstms and cnns. *arXiv preprint arXiv:1903.07288*, 2019.

[46] ECHELON. *IzoT Platform*, 2018 (accessed January 04, 2018). `https://www.echelon.com/izot-platform`.

[47] Rob Enns, Martin Bjorklund, Juergen Schoenwaelder, and Andy Bierman. Network configuration protocol (netconf). 2011.

[48] Zainab G Faisal and Nadia A Shiltagh. Traffic management in wireless sensor network based on modified neural networks. *Iraqi Journal for Computers and Informatics ijci*, 41(1):4–8, 2014.

[49] Shahin Farahani. *ZigBee wireless networks and transceivers*. Newnes, 2011.

[50] Hossam Faris, Ibrahim Aljarah, Mohammed Azmi Al-Betar, and Seyedali Mirjalili. Grey wolf optimizer: a review of recent variants and applications. *Neural computing and applications*, 30(2):413–435, 2018.

[51] Eduardo Feo-Flushing, Michal Kudelski, Jawad Nagi, Luca M Gambardella, and Gianni A Di Caro. Link quality estimationâĂŤa case study for on-line supervised learning in wireless sensor networks. In *Real-World Wireless Sensor Networks*, pages 97–101. Springer, 2014.

[52] DF Jenolin Flora, V Kavitha, and M Muthuselvi. A survey on congestion control techniques in wireless sensor networks. In *2011 International Conference on Emerging Trends in Electrical and Computer Technology*, pages 1146–1149. IEEE, 2011.

[53] Anna Forster. Machine learning techniques applied to wireless ad-hoc networks: Guide and survey. In *2007 3rd international conference on intelligent sensors, sensor networks and information*, pages 365–370. IEEE, 2007.

[54] Anna Förster and Amy L Murphy. Froms: A failure tolerant and mobility enabled multicast routing paradigm with reinforcement learning for wsns. *Ad Hoc Networks*, 9(5):940–965, 2011.

[55] Olfa Gaddour, Anis Koubâa, Nouha Baccour, and Mohamed Abid. Of-fl: Qos-aware fuzzy logic objective function for the rpl routing protocol. In *2014 12th International symposium on modeling and optimization in mobile, ad hoc, and wireless networks (WiOpt)*, pages 365–372. IEEE, 2014.

[56] Vicente García, José Salvador Sánchez, and Ramón Alberto Mollineda. On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. *Knowledge-Based Systems*, 25(1):13–21, 2012.

[57] Ali Ghaffari. Congestion control mechanisms in wireless sensor networks: A survey. *Journal of network and computer applications*, 52:101–115, 2015.

[58] Majid Gholipour, Abolfazl Toroghi Haghighat, and Mohammad Reza Meybodi. Hop-by-hop traffic-aware routing to congestion control in wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2015(1):15, 2015.

[59] Majid Gholipour, Abolfazl Toroghi Haghighat, and Mohammad Reza Meybodi. Hop-by-hop congestion avoidance in wireless sensor networks based on genetic support vector machine. *Neurocomputing*, 223:63–76, 2017.

[60] Khusvinder Gill, Shuang-Hua Yang, Fang Yao, and Xin Lu. A zigbee-based home automation system. *IEEE Transactions on consumer Electronics*, 55(2):422–430, 2009.

[61] Jorge Granjal, Edmundo Monteiro, and Jorge Sá Silva. Security for the internet of things: a survey of existing protocols and open research issues. *IEEE Communications Surveys & Tutorials*, 17(3):1294–1312, 2015.

[62] Guo Haixiang, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue, and Gong Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, 73:220–239, 2017.

[63] Akram Hakiri, Pascal Berthou, Aniruddha Gokhale, and Slim Abdellatif. Publish/subscribe-enabled software defined networking for efficient and scalable iot communications. *IEEE communications magazine*, 53(9):48–54, 2015.

[64] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[65] Ranida Hamidouche, Zibouda Aliouat, Abdelhak Mourad Gueroui, Ado Adamou Abba Ari, and Lemia Louail. Classical and bio-inspired mobility in sensor networks for iot applications. *Journal of Network and Computer Applications*, 121:70–88, 2018.

[66] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.

[67] Mohammad Mehedi Hassan, Biao Song, and Eui-Nam Huh. A framework of sensor-cloud integration opportunities and challenges. In *Proceedings of the 3rd international conference on Ubiquitous information management and communication*, pages 618–626, 2009.

[68] Karel Heurtefeux and Hamid Menouar. Experimental evaluation of a routing protocol for wireless sensor networks: Rpl under study. In *6th Joint IFIP Wireless and Mobile Networking Conference (WMNC)*, pages 1–4. IEEE, 2013.

[69] Ru Huang, Xiaoli Chu, Jie Zhang, and Yu Hen Hu. Energy-efficient monitoring in software defined wireless sensor networks using reinforcement learning: A prototype. *International Journal of Distributed Sensor Networks*, 11(10):360428, 2015.

[70] Ray Hunt. Snmp, snmpv2 and cmip - the technologies for multivendor network management. *Computer Communications*, 20(2):73–88, 1997.

[71] Maab Alaa Hussain. A radial basis neural network controller to solve congestion in wireless sensor networks. *Iraqi Journal for Computers and Informatics ijci*, 44(1):53–62, 2018.

[72] IBM. *IBM Watson IoT Platform*, 2017, (accessed January 04, 2018). `https://internetofthings.ibmcloud.com/#/`.

[73] Oana Iova, Fabrice Theoleyre, and Thomas Noel. Stability and efficiency of rpl under realistic conditions in wireless sensor networks. In *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 2098–2102. IEEE, 2013.

[74] Oana Iova, Fabrice Theoleyre, and Thomas Noel. Using multiparent routing in rpl to increase the stability and the lifetime of the network. *Ad Hoc Networks*, 29:45–62, 2015.

[75] Oana Iova, Fabrice Theoleyre, Mengchuan Zou, and Jia-liang Lu. Efficient and reliable mac-layer broadcast for ieee 802.15. 4 wireless sensor networks. In *2014 7th IFIP Wireless and Mobile Networking Conference (WMNC)*, pages 1–8. IEEE, 2014.

[76] H. Ismail, H. S. Hamza, and S. M. Mohamed. Semantic enhancement for network configuration management. In *2018 IEEE Global Conference on Internet of Things (GCIoT)*, pages 1–5, 2018.

[77] Martin Jacobsson and Charalampos Orfanidis. Using software-defined networking principles for wireless sensor networks. In *SNCNW 2015, May 28–29, Karlstad, Sweden*, 2015.

[78] Anil K Jain, Jianchang Mao, and K Moidin Mohiuddin. Artificial neural networks: A tutorial. *Computer*, 29(3):31–44, 1996.

[79] Aarti Jangid and Parul Chauhan. A survey and challenges in iot networks. In *2019 International Conference on Intelligent Sustainable Systems (ICISS)*, pages 516–521. IEEE, 2019.

[80] Brownlee Jason. *Confidence Intervals for Machine Learning*, 2019, (Accessed June 22, 2020). `https://machinelearningmastery.com/confidence-intervals-for-machine-learning/`.

[81] KAA. *IoT cloud platform the Internet of Things solutions and applications that set the standard*, 2018 (accessed January 04, 2018). `https://www.kaaproject.org/`.

[82] Sophia Kaplantzis, Alistair Shilton, Nallasamy Mani, and Y Ahmet Sekercioglu. Detecting selective forwarding attacks in wireless sensor networks using support vector machines. In *2007 3rd International Conference on Intelligent Sensors, Sensor Networks and Information*, pages 335–340. IEEE, 2007.

[83] Artem Katasonov, Olena Kaykova, Oleksiy Khriyenko, Sergiy Nikitin, and Vagan Y Terziyan. Smart semantic middleware for the internet of things. *Icinco-Icso*, 8:169–178, 2008.

[84] Hafiza Syeda Zainab Kazmi, Nadeem Javaid, Muhammad Awais, Muhammad Tahir, Seong-o Shim, and Yousaf Bin Zikria. Congestion avoidance and fault detection in wsns using data science techniques. *Transactions on Emerging Telecommunications Technologies*, 2019.

[85] Hafiza Syeda Zainab Kazmi, Nadeem Javaid, Muhammad Imran, and Fatma Outay. Congestion control in wireless sensor networks based on support vector

machine, grey wolf optimization and differential evolution. In *2019 Wireless Days (WD)*, pages 1–8. IEEE, 2019.

[86] Mounib Khanafer, Mouhcine Guennoun, and Hussein T Mouftah. A survey of beacon-enabled ieee 802.15. 4 mac protocols in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 16(2):856–876, 2013.

[87] Hyojoon Kim and Nick Feamster. Improving network management with software defined networking. *IEEE Communications Magazine*, 51(2):114–119, 2013.

[88] Kyuhyung Kim, Sungwon Lee, Hongseok Yoo, and Dongkyun Kim. Agriculture sensor-cloud infrastructure and routing protocol in the physical sensor network layer. *International Journal of Distributed Sensor Networks*, 10(3):437535, 2014.

[89] Jeong-Gil Ko, Yong-Hyun Cho, and Hyogon Kim. Performance evaluation of ieee 802.15. 4 mac with different backoff ranges in wireless sensor networks. In *2006 10th IEEE Singapore International Conference on Communication Systems*, pages 1–5. IEEE, 2006.

[90] Brent Komer, James Bergstra, and Chris Eliasmith. Hyperopt-sklearn: automatic hyperparameter configuration for scikit-learn. In *ICML workshop on AutoML*, volume 9. Citeseer, 2014.

[91] Djamel Eddine Kouicem, Abdelmadjid Bouabdallah, and Hicham Lakhlef. Internet of things security: A top-down survey. *Computer Networks*, 141:199–221, 2018.

[92] Martin Kubisch, Holger Karl, Adam Wolisz, Lizhi Charlie Zhong, and Jan Rabaey. Distributed algorithms for transmission power control in wireless sensor networks. In *2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.*, volume 1, pages 558–563. IEEE, 2003.

[93] D Praveen Kumar, Tarachand Amgoth, and Chandra Sekhara Rao Annavarapu. Machine learning algorithms for wireless sensor networks: A survey. *Information Fusion*, 49:1–25, 2019.

[94] Adrian Lara, Anisha Kolasani, and Byrav Ramamurthy. Network innovation using openflow: A survey. *IEEE communications surveys & tutorials*, 16(1):493–512, 2013.

[95] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

[96] Jie Lin, Wei Yu, Nan Zhang, Xinyu Yang, Hanlin Zhang, and Wei Zhao. A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet of Things Journal*, 4(5):1125–1142, 2017.

[97] Hanna Lindholm-Ventola and Bilhanan Silverajan. Coap-snmp interworking iot scenarios. 2014.

[98] Tao Liu and Alberto E Cerpa. Foresee (4c): Wireless link prediction using link features. In *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 294–305. IEEE, 2011.

[99] Zhenzhen Liu and Itamar Elhanany. Rl-mac: a reinforcement learning based mac protocol for wireless sensor networks. *International Journal of Sensor Networks*, 1(3-4):117–124, 2006.

[100] LogMeIn. *IoT Platform for Connected Devices*, 2018 (accessed January 04, 2018). `https://www.xively.com/`.

[101] Jayashri B Madalgi and S Anupama Kumar. Congestion detection in wireless sensor networks using mlp and classification by regression. In *2017 3rd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, pages 226–231. IEEE, 2017.

[102] Omolemo Godwill Matlou and Adnan M Abu-Mahfouz. Utilising artificial intelligence in software defined wireless sensor network. In *IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society*, pages 6131–6136. IEEE, 2017.

[103] Douglas Mauro and Kevin Schmidt. *Essential SNMP: Help for System and Network Administrators*. " O'Reilly Media, Inc.", 2005.

[104] Mahir Meghji and Daryoush Habibi. Transmission power control in multihop wireless sensor networks. In *2011 Third International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 25–30. IEEE, 2011.

[105] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011.

[106] Mihail Mihaylov, Yann-Aël Le Borgne, Karl Tuyls, and Ann Nowé. Decentralised reinforcement learning for energy-efficient scheduling in wireless sensor networks. *International Journal of Communication Networks and Distributed Systems*, 9(3-4):207–224, 2012.

[107] Hadi Mollaei and Azadeh Alsadat Emrani Zarandi. New method for congestion control in wireless sensor network using neural network. *Revista QUID*, 1(1):1085–1093, 2017.

[108] Azzam I Moustapha and Rastko R Selmic. Wireless sensor network modeling using modified recurrent neural networks: Application to fault detection. *IEEE Transactions on Instrumentation and Measurement*, 57(5):981–988, 2008.

[109] Nitin Naik. Choice of effective messaging protocols for iot systems: Mqtt, coap, amqp and http. In *2017 IEEE international systems engineering symposium (ISSE)*, pages 1–7. IEEE, 2017.

[110] Musa Ndiaye, Gerhard P Hancke, and Adnan M Abu-Mahfouz. Software defined networking for improved wireless sensor network management: A survey. *Sensors*, 17(5):1031, 2017.

[111] Ayla Networks. *IoT Software | IoT Platform | Ayla Networks*, 2018 (accessed March 12, 2018). `https://www.aylanetworks.com/`.

[112] Tamoghna Ojha, Samaresh Bera, Sudip Misra, and Narendra Singh Raghuwanshi. Dynamic duty scheduling for green sensor-cloud applications. In *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*, pages 841–846. IEEE, 2014.

[113] Flauzac Olivier, Gonzalez Carlos, and Nolot Florent. Sdn based architecture for clustered wsn. In *2015 9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 342–347. IEEE, 2015.

[114] Charalampos Orfanidis. Ph. d. forum abstract: Increasing robustness in wsn using software defined network architecture. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 1–2. IEEE, 2016.

[115] Fredrik Osterlind, Adam Dunkels, Joakim Eriksson, Niclas Finne, and Thiemo Voigt. Cross-level sensor network simulation with cooja. In *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*, pages 641–648. IEEE, 2006.

[116] Pangun Park, Piergiuseppe Di Marco, Carlo Fischione, and Karl Henrik Johansson. Modeling and optimization of the ieee 802.15. 4 protocol for reliable and timely communications. *IEEE Transactions on Parallel and Distributed Systems*, 24(3):550–564, 2012.

[117] Pangun Park, Piergiuseppe Di Marco, Pablo Soldati, Carlo Fischione, and Karl Henrik Johansson. A generalized markov chain model for effective analysis of slotted ieee 802.15. 4. In *2009 IEEE 6th International Conference on Mobile Adhoc and Sensor Systems*, pages 130–139. IEEE, 2009.

[118] Pangun Park, Sinem Coleri Ergen, Carlo Fischione, and Alberto Sangiovanni-Vincentelli. Duty-cycle optimization for ieee 802.15. 4 wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 10(1):1–32, 2013.

[119] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[120] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[121] Houda Rachidi and Ahmed Karmouch. A framework for self-configuring devices using tr-069. In *2011 International Conference on Multimedia Computing and Systems*, pages 1–6. IEEE, 2011.

[122] Predrag Radivojac, Uttara Korad, Krishna M Sivalingam, and Zoran Obradovic. Learning from class-imbalanced data in wireless sensor networks. In *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall (IEEE Cat. No. 03CH37484)*, volume 5, pages 3030–3034. IEEE, 2003.

[123] Ashfaqur Rahman, Daniel V Smith, and Greg Timms. A novel machine learning approach toward quality assessment of sensor data. *IEEE Sensors Journal*, 14(4):1035–1047, 2013.

[124] Hassan Ramchoun, Mohammed Amine Janati Idrissi, Youssef Ghanou, and Mohamed Ettaouil. Multilayer perceptron: Architecture optimization and training. *IJIMAI*, 4(1):26–30, 2016.

[125] Suhas Rao, Devaiah Chendanda, Chetan Deshpande, and Vishwas Lakkundi. Implementing lwm2m in constrained iot devices. In *2015 IEEE Conference on Wireless Sensors (ICWiSe)*, pages 52–57. IEEE, 2015.

[126] Tifenn Rault, Abdelmadjid Bouabdallah, and Yacine Challal. Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks*, 67:104–122, 2014.

[127] Dawn Rohm, Mukul Goyal, Hossein Hosseini, August Divjak, and Yusuf Bashir. Configuring beaconless ieee 802.15. 4 networks under different traffic loads. In *2009 International Conference on Advanced Information Networking and Applications*, pages 921–928. IEEE, 2009.

[128] Kristina Sahlmann, Thomas Scheffler, and Bettina Schnor. Managing iot device capabilities based on onem2m ontology descriptions. *Proceedings of the 16. GI/ITG KuVS Fachgespräch Sensornetze (Technical Reports)*, 2017.

[129] Peter Saint-Andre. Extensible messaging and presence protocol (xmpp): Address format. Technical report, RFC 6122, March, 2011.

[130] SAMSUNG. *IoT Cloud Platform - Samsung ARTIK cloud services*, 2017 (accessed March 12, 2018). `https://artik.cloud/`.

[131] Thomas Scheffler and Olaf Bonneß. Manage resource-constrained iot devices through dynamically generated and deployed yang models. In *Proceedings of the Applied Networking Research Workshop*, pages 42–47, 2017.

[132] Mischa Schmidt, Shahd Safarani, Julia Gastinger, Tobias Jacobs, Sébastien Nicolas, and Anett Schülke. On the performance of differential evolution for hyperparameter tuning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[133] J Schoenwaelder, K Watsen, M Ersue, and V Perelman. Network configuration protocol light (netconf light). In *Working Draft, IETF Secretariat, Internet-Draft draft-schoenw-netconf-light-01*, 2012.

[134] Pallavi Sethi and Smruti R Sarangi. Internet of things: architectures, protocols, and applications. *Journal of Electrical and Computer Engineering*, 2017, 2017.

[135] Sakir Sezer, Sandra Scott-Hayward, Pushpinder Kaur Chouhan, Barbara Fraser, David Lake, Jim Finnegan, Niel Viljoen, Marc Miller, and Navneet Rao. Are we ready for sdn? implementation challenges for software-defined networks. *IEEE Communications Magazine*, 51(7):36–43, 2013.

[136] Kunal Shah and Mohan Kumar. Distributed independent reinforcement learning (dirl) approach to resource management in wireless sensor networks. In *2007 IEEE International Conference on Mobile Adhoc and Sensor Systems*, pages 1–9. IEEE, 2007.

[137] Alexander Shalimov, Dmitry Zuikov, Daria Zimarina, Vasily Pashkov, and Ruslan Smeliansky. Advanced study of sdn/openflow controllers. In *Proceedings of the 9th central & eastern european software engineering conference in russia*, pages 1–6, 2013.

[138] Zhengguo Sheng, Chinmaya Mahapatra, Chunsheng Zhu, and Victor CM Leung. Recent advances in industrial wireless sensor networks toward efficient management in iot. *IEEE access*, 3:622–637, 2015.

[139] Zhengguo Sheng, Hao Wang, Changchuan Yin, Xiping Hu, Shusen Yang, and Victor CM Leung. Lightweight management of resource-constrained sensor devices in internet of things. *IEEE internet of things journal*, 2(5):402–411, 2015.

[140] Zhengguo Sheng, Shusen Yang, Yifan Yu, Athanasios V Vasilakos, Julie A McCann, and Kin K Leung. A survey on the ietf protocol suite for the

internet of things: Standards, challenges, and opportunities. *IEEE wireless communications*, 20(6):91–98, 2013.

[141] Soraya Sinche, Duarte Raposo, Ngombo Armando, André Rodrigues, Fernando Boavida, Vasco Pereira, and Jorge Sá Silva. A survey of iot management protocols and frameworks. *IEEE Communications Surveys & Tutorials*, 22(2):1168–1190, 2019.

[142] Prakul Singhal and Anamika Yadav. Congestion detection in wireless sensor network using neural network. In *International Conference for Convergence for Technology-2014*, pages 1–4. IEEE, 2014.

[143] SiteWhere. *The Open Platform for the Internet of Things*, 2018 (accessed January 04, 2018). `http://www.sitewhere.org/`.

[144] Chao Song, Ming Liu, Jiannong Cao, Yuan Zheng, Haigang Gong, and Guihai Chen. Maximizing network lifetime based on transmission range adjustment in wireless sensor networks. *Computer Communications*, 32(11):1316–1325, 2009.

[145] Giorgio Stampa, Marta Arias, David Sánchez-Charles, Victor Muntés-Mulero, and Albert Cabellos. A deep-reinforcement learning approach for software-defined networking routing optimization. *arXiv preprint arXiv:1709.07080*, 2017.

[146] Andy Stanford-Clark and Hong Linh Truong. Mqtt for sensor networks (mqtt-sn) protocol specification. *International business machines (IBM) Corporation version*, 1:2, 2013.

[147] George Suciu, Alexandru Vulpe, Simona Halunga, Octavian Fratu, Gyorgy Todoran, and Victor Suciu. Smart cities built on resilient cloud computing and secure internet of things. In *2013 19th international conference on control systems and computer science*, pages 513–518. IEEE, 2013.

[148] Prodea Systems. *Prodea's Arrayent IoT Services Platform*, 2018 (accessed March 12, 2018). `https://prodea.com/iot-services-platform/`.

[149] Li Qiang Tao and Feng Qi Yu. Ecoda: enhanced congestion detection and avoidance for multiple class of traffic in sensor networks. *IEEE transactions on consumer electronics*, 56(3):1387–1394, 2010.

[150] thethings.io. *The IoT platform to monitorize your devices*, 2018 (accessed January 04, 2018). `https://thethings.io/`.

[151] thinger.io. *Open Source IoT Platform*, 2018 (accessed January 04, 2018). `https://thinger.io/`.

[152] ThingWorx. *ThingWorx Industrial IoT Platform*, 2018 (accessed March 12, 2018). `https://www.ptc.com/en/products/iot/thingworx-platform`.

[153] Dan Tudose, Laura Gheorghe, and N Tapus. Radio transceiver consumption modeling for multi-hop wireless sensor networks. *UPB Scientific Bulletin, Series C*, 75(1):17–26, 2013.

[154] András Varga and Rudolf Hornig. An overview of the omnet++ simulation environment. In *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, page 60. ICST (Institute for Computer Sciences, Social-Informatics and âĂę, 2008.

[155] Michel Veillette, Peter Van der Stok, Alexander Pelov, Andy Bierman, and Ivaylo Petrov. CoAP Management Interface. Internet-Draft draft-ietf-core-comi-09, Internet Engineering Task Force, March 2020. Work in Progress.

[156] Panagiotis Vlacheas, Raffaele Giaffreda, Vera Stavroulaki, Dimitris Kelaidonis, Vassilis Foteinos, George Poulios, Panagiotis Demestichas, Andrey Somov, Abdur Rahim Biswas, and Klaus Moessner. Enabling smart cities through a cognitive management framework for the internet of things. *IEEE communications magazine*, 51(6):102–111, 2013.

[157] Chieh-Yih Wan, Shane B Eisenman, and Andrew T Campbell. Coda: Congestion detection and avoidance in sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 266–279, 2003.

[158] Feng Wang, Dou Li, and Yuping Zhao. Analysis and compare of slotted and unslotted csma in ieee 802.15. 4. In *2009 5th International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1–5. IEEE, 2009.

[159] Guangxue Wang and Kai Liu. Upstream hop-by-hop congestion control in wireless sensor networks. In *2009 IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1406–1410. IEEE, 2009.

[160] Mowei Wang, Yong Cui, Xin Wang, Shihan Xiao, and Junchen Jiang. Machine learning for networking: Workflow, advances and opportunities. *Ieee Network*, 32(2):92–99, 2017.

[161] Ping Wang and Ting Wang. Adaptive routing for sensor networks using reinforcement learning. In *The Sixth IEEE International Conference on Computer and Information Technology (CIT'06)*, pages 219–219. IEEE, 2006.

[162] Yong Wang, Margaret Martonosi, and Li-Shiuan Peh. A supervised learning approach for routing optimizations in wireless sensor networks. In *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*, pages 79–86, 2006.

[163] Yong Wang, Margaret Martonosi, and Li-Shiuan Peh. Predicting link quality using supervised learning in wireless sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 11(3):71–83, 2007.

[164] K Watsen and RESTCONF Protocol. Network working group a. bierman internet-draft yumaworks intended status: Standards track m. bjorklund expires: April 30, 2017 tail-f systems. 2016.

[165] Tim Winter. Rpl: Ipv6 routing protocol for low-power and lossy networks. 2012.

[166] Jun Wu, Kaoru Ota, Mianxiong Dong, and Chunxiao Li. A hierarchical security framework for defending against sophisticated attacks on wireless sensor networks in smart cities. *IEEE Access*, 4:416–424, 2016.

[167] Yi Xu and Abdelsalam Helal. Scalable cloud–sensor architecture for the internet of things. *IEEE Internet of Things Journal*, 3(3):285–298, 2015.

[168] Hang Yang, Simon Fong, Raymond Wong, and Guangmin Sun. Research article timi ing classification ecision rees by using weighted naïve bayes predictors to reduce the imbalanced class problem in wireless sensor network. 2013.

[169] Xin Yang, Jianlin Guo, Philip Orlik, Kieran Parsons, and Koichi Ishibashi. Stability metric based routing protocol for low-power and lossy networks. In *2014 IEEE International Conference on Communications (ICC)*, pages 3688–3693. IEEE, 2014.

[170] Ibrar Yaqoob, Ejaz Ahmed, Ibrahim Abaker Targio Hashem, Abdelmuttlib Ibrahim Abdalla Ahmed, Abdullah Gani, Muhammad Imran, and Mohsen Guizani. Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges. *IEEE wireless communications*, 24(3):10–16, 2017.

[171] Guanhua Ye, Tarek N Saadawi, and Myung Lee. On explicit congestion notification for stream control transmission protocol in lossy networks. *Cluster Computing*, 8(2-3):147–156, 2005.

[172] Halil Yetgin, Kent Tsz Kan Cheung, Mohammed El-Hajjar, and Lajos Hanzo Hanzo. A survey of network lifetime maximization techniques in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 19(2):828–854, 2017.

[173] Jun Min Yi, Eom Ji Oh, Dong Kun Noh, and Ikjune Yoon. Energy-aware data compression and transmission range control for energy-harvesting wireless sensor networks. *International Journal of Distributed Sensor Networks*, 13(4):1550147717705785, 2017.

[174] Xiaoyan Yin, Xingshe Zhou, Rongsheng Huang, Yuguang Fang, and Shining Li. A fairness-aware congestion control scheme in wireless sensor networks. *IEEE transactions on vehicular technology*, 58(9):5225–5234, 2009.

[175] Madoka Yuriyama and Takayuki Kushida. Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing. In *2010 13th International Conference on Network-Based Information Systems*, pages 1–8. IEEE, 2010.

[176] Maciej Zawodniok and Sarangapani Jagannathan. Predictive congestion control protocol for wireless sensor networks. *IEEE Transactions on Wireless Communications*, 6(11):3955–3963, 2007.

[177] Ruifeng Zhang and Jean-Marie Gorce. Optimal transmission range for minimum energy consumption in wireless sensor networks. In *2008 IEEE*

Wireless Communications and Networking Conference, pages 757–762. IEEE, 2008.