Machine Learning for Automatic Machine-to-Machine (M2M) Communication in 5G

Seifeddine Messaoud

RÉPUBLIQUE TUNISIENNE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉERIEUR

ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ DE MONASTIR

FACULTÉ DES SCIENCES DE MONASTIR

Ecole Doctorale : Matériaux, Dispositifs et Microsystèmes (EDMD)

THÈSE

Présentée pour l'obtention du diplôme de

DOCTORAT DE L'UNIVERSITÉ DE MONASTIR

Faculté des Sciences de Monastir

Spécialité : ÉLECTRONIQUE ET MICROÉLECTRONIQUE

Présentée par:

Seifeddine MESSAOUD

Sujet:

Apprentissage Automatique pour la Communication Machine à Machine (M2M) Autonome dans la 5G.

Machine Learning for Autonomic Machine-to-Machine (M2M) Communication in 5G.

Directeur de thèse : **Mohamed ATRI**

Co-directeur de thèse : **Abbas BRADAI**

Soutenue le 18/01/2021 devant la Commission d'Examen composée de :

M. Kamel BESBES	Professeur, CRMN	Président
M. Noureddine LIOUANE	Professeur, ENIM	Rapporteur
M. Noureddine BOULEJFEN	Professeur, CRM	Rapporteur
M. Belgacem HAMDI	Professeur, ISSAT	Examinateur
M. Mohamed ATRI	Professeur, FSM	Directeur de Thèse
M. Abbas BRADAI	Maître de conférences, XLIM	Co-directeur de Thèse

Laboratoire d'Électronique et Microélectronique (Code: LR99ES30)

RÉPUBLIQUE TUNISIENNE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉERIEUR

ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ DE MONASTIR

FACULTÉ DES SCIENCES DE MONASTIR

Ecole Doctorale : Matériaux, Dispositifs et Microsystèmes (EDMD)

THÈSE

Présentée pour l'obtention du diplôme de

DOCTORAT DE L'UNIVERSITÉ DE MONASTIR

Faculté des Sciences de Monastir

Spécialité : ÉLECTRONIQUE ET MICROÉLECTRONIQUE

Présentée par:

Seifeddine MESSAOUD

Sujet:

Apprentissage Automatique pour la Communication Machine à Machine (M2M) Autonome dans la 5G.

Machine Learning for Autonomic Machine-to-Machine (M2M) Communication in 5G.

Directeur de thèse : **Mohamed ATRI**

Co-directeur de thèse : **Abbas BRADAI**

Soutenue le 18/01/2021 devant la Commission d'Examen composée de :

M. Kamel BESBES	Professeur, CRMN	Président
M. Noureddine LIOUANE	Professeur, ENIM	Rapporteur
M. Noureddine BOULEJFEN	Professeur, CRM	Rapporteur
M. Belgacem HAMDI	Professeur, ISSAT	Examinateur
M. Mohamed ATRI	Professeur, FSM	Directeur de Thèse
M. Abbas BRADAI	Maître de conférences, XLIM	Co-directeur de Thèse

Laboratoire d'Électronique et Microélectronique (Code: LR99ES30)

Acknowledgments

First, I would like to witness my gratitude to Almighty God, who gave me the courage and strength to pursue this thesis and opened the gates of knowledge for me.

I have started my final internship project for my master's degree in Electronics and Microelectronics Lab Research EμE. It was March 2017. I never thought this training would be the reason for choosing me to pursue a long and fruitful Ph.D. journey full of fluctuations (The 1st registration was on January 4, 2018). What's for sure is that I will never forget this opportunity which was wonderful, overwhelming, and full of lessons and responsibilities. Thankfully, my Ph.D. was completed successfully in which we delivered many journals and international conferences. This wouldn't have happened without the aid and support of countless people over the past three years.

First of all, I am deeply grateful to my advisors ***Mr. Mohamed ATRI*** *and* ***Mr. Abbas BRADAI*** *who gave me this opportunity and believed in me from the very beginning. Knowing that this Ph.D. experience wasn't always easy, it was always a pleasure, with a lot of excitement, to be up to the challenge and to beat paper deadlines. Working with you improved me a lot as a student, as a researcher, and as a person. It wasn't straightforward for me to understand how to think like a researcher. You have taught me, both consciously and unconsciously, how good work is done. I appreciate all your contributions of time and ideas to make my Ph.D. experience productive. You have always listened to my ideas and discussions with you frequently led to progress. Your ability to approach research problems and your high scientific standards set an example. I admire your ability to balance research interests and personal pursuits. I am thankful for the excellent example you have provided me as a successful and ambitious researcher.*

I would first of all like to thank the members of the jury for their presence, for their careful reading of my thesis as well as for the remarks they will address to me during

this defense in order to improve my work. I thank Professors ***Noureddine LIOUANE*** *and* ***Noureddine BOULEJFEN*** *for finding the time to read my thesis and for their valuable feedback. I would like to thank Professor* ***Belgacem HAMDI*** *for being examinator of this jury. And finally, I would like to thank Professor* ***Kamel BESBES*** *for the honor he gave me when he agreed to chair this jury.*

I would like to thank all EμE laboratory members, especially the Lab head Pr. ***Mohsen MACHHOUT****, and I have been very privileged to get support from many great people who became friends over the last several years. In addition, I would like to thank all XLIM laboratory members (the lab where I did my internship and collaboration), but one special thanks go to my friend Samir DAWALIBY, for always being by my side during tough times and for his continuous encouragement and especially when I was in France.*

I would like to thank my dear lovely mother in the world, ***Dalila****, and my dear soulful father* ***Mohamed*** *and the big advisor for me. Also, I thank my dear lovely sister* ***Imen*** *and my dear feisty brother* ***Hazem*** *for their illimitable love. Your encouragements and your faiths in me gave me the strength to achieve my goal.*

Seifeddine MESSAOUD

“ “ “ “ “ “

Abstract

Fifth Generation (5G) is envisioned by Telecom operators as the next mobile network generation which brings solutions to massive customers who are more and more demanding for both cost-efficient services as well as high satisfaction. Machine to machine (M2M) communication is an important component of the future 5G, which enables the ubiquitous connectivity between a myriad of machines without or with limited human intervention. Thus, the autonomous connection of devices facilitates the emergence of a wide range of intelligent M2M applications. These latter have exhibited a strong potential to improve human life in different fields such as eHealth, smart grids, smart home/cities, intelligent transportation and surveillance, enabling partially the internet of things (IoT). By 2020, the number of connected devices is expected to reach 50 billions. To meet these huge demands, the network autonomic mechanisms, named as "Autonomic Networking", are considered as good candidates to provide intelligent networking solutions for M2M communication in 5G networks. The objective of this thesis is to improve the self-organization and autonomic networking approaches with the integration of machine learning and data analytics for M2M communication. This approach based on machine learning and data analysis will be used to customize the QoS for users through the customization of Autonomic Networking.

Résumé

La cinquième génération (5G) est envisagée par les opérateurs de télécommunications comme la prochaine génération de réseaux mobiles qui apporte des solutions à des clients massifs qui sont de plus en plus exigeants à la fois pour des services rentables et pour une grande satisfaction. La communication de machine à machine (M2M) est un élément important de la future 5G, qui permet la connectivité omniprésente entre une myriade de machines sans ou avec une intervention humaine limitée. Ainsi, la connexion autonome des appareils facilite l'émergence d'une large gamme d'applications M2M intelligentes. Ces derniers ont montré un fort potentiel pour améliorer la vie humaine dans différents domaines tels que la cybersanté, les réseaux intelligents, la maison/Les villes intelligentes, les transports intelligents et la surveillance, permettant l'Internet des objets (IoT). D'ici 2020, le nombre d'appareils connectés devrait atteindre 50 milliards. Pour répondre à ces énormes demandes, les mécanismes autonomes de réseau, appelés «Autonomic Networking», sont considérés comme de bons candidats pour fournir des solutions de réseau intelligentes pour la communication M2M dans les réseaux 5G. L'objectif de cette thèse est d'améliorer les approches d'auto-organisation et de réseautage autonome avec l'intégration de l'apprentissage automatique et de l'analyse de données pour la communication M2M. Cette approche basée sur l'apprentissage automatique et l'analyse des données sera utilisée pour personnaliser la qualité de service (QdS) pour les utilisateurs grâce à la personnalisation du réseau autonome.

Introduction Générale

Avec le développement des réseaux sans fil de cinquième génération (5G), on s'attend d'ici 2022 à voir s'établir une connexion efficace entre les humains et les machines afin de garantir la flexibilité nécessaire pour gérer des réseaux avec des besoins en qualité de service hétérogènes. Actuellement, les opérateurs font face à divers défis et challenges lors du déploiement des communications IoT à travers les réseaux existants. Inévitablement, l'augmentation du nombre de dispositifs IoT connectés pose des problèmes de charge et de congestion et auront un impact important sur les systèmes de communication sans fil. Les dispositifs IoT nécessitent principalement une batterie de longue durée de vie, une portée étendue, une capacité plus grande pour prendre en charge des millions de dispositifs avec un coût de déploiement faible. Pour répondre à ces caractéristiques, plusieurs technologies ont été proposées et développées pour assurer la meilleure efficacité des réseaux étendus à faible consommation énergétique (LPWAN).

L'objectif du chapitre I est de présenter les technologies LPWAN existant actuellement. Une partie de ces technologies fonctionne dans un spectre de fréquences libre LoRa et Sigfox. Nous avons commencé par évaluer la performance deces technologies en se basant sur des études de littératures. En se basant sur cette evaluation, nous avons choisi exploiter la technology LoRaWAN qui supporte un réseau IoT plus condensé (des milliers de dispositifs IoT peuvent être simulés dans une seule cellule). LoRa commence à être de plus en plus répandu vu son accessibilité basée sur un code source ouvert contrairement à Sigfox qui est plutôt une technologie propriétaire. Ensuite, nous mettons l'accent sur les derniers travaux de recherche visant à optimiser les communications IoT et la gestion des ressources à l'aide des nouvelles technologies qui assurent la virtualisation et la programmabilité des réseaux IoT. Cependant, en partant de l'état de l'art, il y a eu peu de travaux de recherche qui se sont focalisés sur la QdS des communications IoT en termes de respect des délais critiques, de garantie d'un certain débit, et d'un taux de réception des paquets élevé. Ces différents réseaux logiques sont appelés des slices du réseau dont chaque slice correspond à un réseau virtuel de bout en bout entre un noeud IoT et un service réseau en s'appuyant sur la même infrastructure réseau

physique. Étant donné que le nombre de périphériques IoT connectés augmente rapidement avec le temps, une solution efficace pour garantir la qualité de service consiste à apporter de la flexibilité et une virtualisation des réseaux IoT à l'aide de SDN et du découpage en slices. Cette QdS sera garantie en favorisant les communications urgentes, et une gestion flexible du réseau divisé en plusieurs réseaux virtuels configurés et gérés séparément. Pour chaque slice, une partie des ressources physiques est réservée de bout en bout sur toutes les couches (accès réseau, coeur et cloud) pour répondre aux besoins QdS des applications urgentes et fiables. Meanwhile, dans ce chapitre, nous mettons l'accent sur l'intelligence artificielle et l'industrie de IoT, et leur combinaison qui vise à améliorer la mise en réseaux et à augmenter la fiabilité tout en augmentant la QdS. Dans cette thèse, afin de pouvoir garantir cette QdS pour les communications IoT, nous devrons d'abord répondre aux questions suivantes::

- Comment affecter les noeuds IoT aux slices et comment classifier ces slices dans le réseau ?
- Comment réserver les ressources physiques de réseau pour chaque slice et à l'intérieur de chaque slice, comment assurer une allocation optimisée des canaux ?
- Quels sont les paramètres qui impactent la QdS de chaque dispositif connecté et comment optimiser la configuration sans augmenter la complexité du réseau et sans impacter sa performance ?
- Les architectures centralisées actuelle sera-elle capable de supporter l'utilisation à grande échelle des communications, et comment pourra t-elle suivre les avancées à venir ?

Dans le chapitre II, le découpage du réseau est implémenté et son efficacité est étudiée sur diverses stratégies de découpage. La première étape consiste à associer et à détacher un nœud IoT d'une tranche de réseau pendant chaque intervalle de temps. Nous proposons de réaliser dynamiquement ce mécanisme avec une méthode basée sur l'algorithme OGMMC qui regroupe les nœuds en fonction du taux d'urgence calculé comme le rapport entre le délai de paquet instantané et le seuil de délai maximum à ne pas dépasser.

Le résultat de cette première partie est un groupe de nœuds associés aux tranches de réseau virtuel. La deuxième étape consiste à trouver une stratégie optimale de réservation de ressources qui réserve des canaux physiques sur chaque passerelle en supposant que le SDN a une vue globale du réseau et des besoins de chaque nœud en termes de débit. Ici, deux stratégies de découpage dynamique sont proposées basées sur la descente de gradient de mini-batch (MBGD) basée sur l'erreur quadratique moyenne (MSE) qui s'adaptent dynamiquement aux exigences de débit de chaque membre de tranche. Après cela, la troisième et dernière étape consiste à allouer des ressources de canal à l'intérieur de chaque tranche virtuelle en classant les appareils IoT en fonction de la valeur d'utilité

calculée en fonction de l'urgence du retard et du taux de congestion. Enfin, le serveur alloue chaque périphérique au canal fournissant la valeur d'utilité la plus élevée (Max-Utility). Les résultats ont montré l'utilité de l'isolation du réseau virtuel pour éviter la dégradation des performances du réseau due au nombre croissant d'appareils IoT dans une tranche plutôt qu'une autre.

Bien que le découpage du réseau améliore les résultats de la qualité de service, il est encore possible d'améliorer les résultats de fiabilité des tranches de réseau. Cependant, la nature de l'accès aléatoire dans le réseau IoT incite à optimiser le découpage du réseau avec une configuration de paramètres basée sur des tranches qui traite chaque tranche virtuelle différemment sans considérer tous les appareils IoT comme des appareils appartenant au même réseau LoRa. L'objectif du Chapitre III est d'améliorer la qualité de service des appareils IoT et de limiter les interférences et les collisions dans chaque réseau virtuel LoRa. Ce chapitre prolonge les contributions précédentes en évaluant en profondeur diverses stratégies de découpage dans des scénarios industriels 4.0 réalistes à grande échelle.

Plus précisément, une comparaison sera faite entre l'estimation dynamique et les algorithmes basés sur la prédiction. Le premier considère de descente de gradient en mini-lot (MBGD) pour la prédiction dynamique et le second est basé sur l'algorithme l'estimation du maximum de vraisemblance (MLE). Nous incluons la QoS dans la méthode d'optimisation multi-objectifs LoRa, qui était auparavant considérée comme une technologie au meilleur effort, dans le but de tester la flexibilité que le découpage de réseau offre en termes de gestion du trafic et d'intégration de QoS. Ensuite, nous appliquons la meilleure stratégie de découpage trouvée dans le chapitre **??**, où la bande passante est efficacement réservée sur chaque LoRa GW séparément en fonction de la prédiction MBGD.

Le but ici est d'éviter la famine des canaux tout en considérant dynamiquement le besoin exact de chaque tranche en commençant par celle avec la priorité de découpage la plus élevée. Cette stratégie sera ensuite comparée au schéma MLE. Nous adaptons le TOPG comme un schéma d'optimisation de découpage basé sur la technique pour l'ordre de préférence par similarité à solution idéale (TOPSIS) et la méthode de la moyenne géométrique (GMM). La méthode proposée configure efficacement les paramètres LoRa SF et TP et améliore les performances de chaque tranche en termes de QoS, de fiabilité et de consommation d'énergie. Ce dernier prend en compte les exigences de QoS de chaque appareil et configure ses paramètres en conséquence. Dans chaque tranche, l'algorithme recherche la combinaison optimale des configurations SF et TP pour chaque périphérique LoRa au lieu d'utiliser le mécanisme ADR qui force la configuration d'un nœud par l'une des distributions répertorié dans **Table. I.3**. La configuration choisie pour un appareil évite les interférences et augmente la probabilité de réussir à recevoir et à décoder le paquet au niveau de la passerelle tout en prenant en compte les autres paquets reçus simultanément.

Les résultats montrent une amélioration majeure en termes de qualité de service et de consommation d'énergie cependant, il est prévu que le nombre d'appareils augmentera avec le temps et dépassera la capacité maximale qui peut actuellement être prise en charge par une passerelle qui, à son tour, augmentera le taux de perte de paquets dans différentes tranches de réseau. Ce défi nous a motivés à chercher une réponse et à trouver une solution pour relever les défis d'évolutivité et garantir la qualité de service et l'efficacité énergétique des appareils IoT.

Malgré l'amélioration des résultats obtenus grâce aux contributions précédentes et sachant que plus de 30 milliards d'appareils IoT sont estimés être connectés dans les réseaux IoT de la future génération d'ici 2022, l'évolutivité est restée un défi important pour les réseaux IoT de nouvelle génération. Dans un scénario de découpage de réseau LoRa, l'architecture centralisée actuelle de l'état de l'art ne sera pas en mesure de relever les défis de la gestion des ressources réseau à venir dans les futurs déploiements IoT à grande échelle. Par conséquent, dans le Chapitre IV, nous proposons une architecture basée sur SDN fédéré (DFQL) qui répond aux défis d'évolutivité en traitant les données et la prise de décision aux multi-agents LoRa (MAQL) (basés sur l'apprentissage par renforcement) à la périphérie du réseau. Le rôle de découpage de la prise de décision et de la configuration du réseau est mis à profit à la passerelle qui définit la stratégie de découpage et les ressources locales qui doivent être réservées après une phase de fédération avec les autres passerelles proches (agents) via l'orchestrateur fédéré global. En se rapprochant de la périphérie du réseau, la combinaison du découpage du réseau avec les avantages du SDN, l'outil de fédération et la technique d'apprentissage par renforcement améliore la fiabilité des communications dans un réseau à grande échelle grâce à son adaptation rapide aux changements dans un environnement IoT industriel encombré.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

5G : Fifth Generation v–vii, xix

ADR : Adaptive Data Rate ix, xxi, 9, 52

API : Application Programming Interface 11

AR : Augmented reality 15

BW : Bandwidth 7

CaF : Carrier Frequency 6

CPSs : Cyber-physical systems 15

CR : Coding Rate 8

CRC : Cyclic Redundancy Check 4

CSS : Chirp Spread Spectrum 4

DBPSK : Differential Binary Phase Shift Keying 3

DFL : Deep Federated Learning 81

DFQL : Deep Federated Q-Learning x, xxii, 75, 85, 100

DQL : Deep Q-Learning 81

EC : Error Correction 8

EE : Energy Efficiency 78

FEC : Forward Error Correction 4

GMM : Geometric Mean Method ix, xxi, 50, 61

HCLE : High-Critical of Latency and Efficiency 14, 50, 76

ICT : Information and communication technologies 15

IIoT : Industrial Internet of Things 16, 24, 75

IoT : Internet of Services 15

IoT : Internet of Things v, vi, xx, 22

LCLE : Low-Critical of Latency and Efficiency 14, 50, 76

LoRa : Long Range vii, xix, 4

LoRaWAN : Long Range Radio Wide Area Network 3, 4

LPWAN : Low Power Wide Area Network vii, xix, 2, 21, 22

M2M : Machine-to-machine v, vi

MAC : Medium Access Control 4

MANO : Management and Orchestration 27, 76

MAQL : Multi-Agent Q-Learning x, xxii, 75, 81, 100

Max-Utility : max utility algorithm viii, xxi, 26, 32

MBGD : Mini-Batch Gradient Descent viii, ix, xxi, 26, 32, 35, 49, 56, 99

MCE : Management and Control Entity 27

MDP : Markov Decision Process 83

ML : Machine Learning 17

MLE : Maximum Likelihood Estimation ix, xxi, 56, 99

MSE : Mean Square Error viii, xxi, 99

NFV : Network Function Virtualization 27, 75

NS : Network Slicing 10

OGMMC : Online Guassian Misture Model Clustering viii, xx, 26, 32

P2P : Point-to-Point 3

PDR : Packet Delivery Ratio 9

QdS : Qualité de Service vi, vii

QoS : Quality of Service v, xx, 27, 32, 75

REL : Reliability 79

RF : Radio Frequency 9, 52

SDN : Software Defined Networking vii, xx, 10, 27, 75, 81

SF : Spreading Factor ix, xxi, 5, 78, 99

Sigfox : Sigfox vii, xix, 2

SINR : Signal to Interference Noise Ratio 7

TOA : Time on Air 9, 52

TOPG : TOPSIS and GMM ix, xxi, 50, 56, 99

TOPSIS : Technique for Order of Preference by Similarity to Ideal Solution ix, xxi, 50, 61

TP : Transmission Power ix, xxi, 6, 78, 99

UCLE : Ultra-Critical of Latency and Efficiency 13, 50, 76

UNB : Ultra Narrow Band 3

VNF : Virtual Network Function 27

General Introduction

With the development of fifth-generation wireless networks (5G), it is expected by 2022 to see an efficient connection between humans and machines to ensure the flexibility needed to manage networks. with heterogeneous quality of service needs. Currently, operators face various challenges and challenges when deploying IoT communications across existing networks. Inevitably, the increase in the number of connected IoT devices poses load and congestion issues and will have a big impact on wireless communication systems. IoT devices mainly require long battery life, extended range, the larger capacity to support millions of devices with low deployment cost. To meet these characteristics, several technologies have been proposed and developed to ensure the best efficiency of wide-area networks with low energy consumption (LPWAN).

The objective of chapter I is to present the LPWAN technologies currently existing. Some of these technologies operate in a free LoRa and Sigfox frequency spectrum. We started by evaluating the performance of these technologies based on literature studies. Based on this evaluation, we have chosen to exploit LoRaWAN technology which supports a more condensed IoT network (thousands of IoT devices can be simulated in a single cell). LoRa is starting to be more and more widespread given its accessibility based on open source code, unlike Sigfox which is rather a proprietary technology. Next, we focus on the latest research aimed at optimizing IoT communications and resource management using new technologies that provide virtualization and programmability of IoT networks.

However, starting from the state of the art, there has been little research that has focused on the QoS of IoT communications in terms of meeting critical deadlines, ensuring a certain throughput, and a high packet reception rate. These different logical networks are called network slices, each slice of which corresponds to an end-to-end virtual network between an IoT node and a network service relying on the same physical network infrastructure. Since the number of connected IoT devices grows rapidly over time, an effective solution to ensure the QoS is to provide flexibility and virtualization of IoT networks using SDN and slicing.

The QoS will be guaranteed by promoting urgent communications, and flexible management of the network divided into several virtual networks configured and modified separately. For each slice, part of the physical resources is reserved from end to end on all layers (network access, core, and cloud) to meet the QoS needs of urgent and reliable applications. Meanwhile, in this chapter, we focus on artificial intelligence and the IoT industry, and their combination which aims to improve networking and increase reliability while increasing QoS. In this thesis, in order to be able to guarantee this QoS for IoT communications, we will first have to answer the following questions:

- How to assign IoT nodes to slices and how to classify these slices in the network?
- How to reserve the physical network resources for each slice and inside each slice, how to ensure an optimized allocation of the channels?
- What are the parameters that impact the QoS of each connected device and how to optimize the configuration without increasing the complexity of the network and without impacting its performance?
- Will the current centralized architectures are able to support the large-scale use of communications, and how will it be able to keep up with future advances?

In Chapter II, network slicing is implemented and its efficiency is investigated over various slicing strategies. The first step involves associating and detaching an IoT node from a network slice during each time interval. We propose to dynamically realize this

mechanism with a method based on OGMMC algorithm which groups the nodes based on the urgency rate computed as the ratio between the instantaneous packet delay and the maximum delay threshold that should not be exceeded.

The result of this first part is a group of nodes associated with the virtual network slices. The second step is to find an optimal resource reservation strategy which reserves physical channels on each gateways assuming that SDN has a global view of the network and the need of each node in terms of throughput. Here, two dynamic slicing strategy are proposed based on Mini-Batch Gradient Descent (MBGD) based on Mean Square Error (MSE) which dynamically adapt to throughput requirements of each slice members. After that, the third and final step is to allocate channel resources inside each virtual slice by classifying IoT devices according to the utility value computed based on delay urgency and congestion rate. Finally, the server allocates each device to the channel providing the highest utility value (Max-Utility). Results have shown the utility of virtual network isolation to avoid network performance degradation that comes from the increasing number of IoT devices in a slice over another.

Although network slicing improved QoS results, there was still a room for improving reliability results in network slices. However, the random-based access nature in IoT network gives the motivation to optimize network slicing with a slice-based parameter configuration that treats each virtual slice differently without considering all IoT devices as devices belonging to the same LoRa network. The goal behind in Chapter III, is to improve QoS of IoT devices and limit interference and collisions in each LoRa virtual network. This chapter extends the previous contributions by in-depth evaluating various slicing strategies in large scale realistic industry 4.0 scenarios. More precisely, a comparison will be made between the dynamic prediction, and estimation-based algorithms. The former considers Maximum Likelihood Estimation (MLE) and the latter is based on Mini-batch Gradient Descent (MBGD) algorithm. We include QoS in LoRa multi-objective optimization method, which was previously considered as a best-effort technology, with the goal to test the flexibility that network slicing provides in terms of traffic management and QoS integration. Then, we apply the best slicing strategy found

in Chapter II, where the bandwidth is efficiently reserved on each LoRa GW separately based on MBGD prediction. The goal here is to avoid channels starvation while considering dynamically the exact need of each slice starting by the one with the highest slicing priority. This, strategy will be after compared to the MLE scheme. We adapt the TOPG as a slicing optimization scheme based on Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) and Geometric Mean Method (GMM). The proposed method efficiently configures LoRa SF and TP parameters and improves the performance of each slice in terms of QoS, reliability and energy consumption. The latter takes into account QoS requirements of each device and configures its parameters accordingly. In each slice, the algorithm searches for the optimal combination of SF and TP configuration for each LoRa device instead of using the ADR mechanism which forces the configuration of a node by one of the distributions listed in **Table. I.3**. The chosen configuration for a device avoids interference and increases the probability of successfully receiving and decoding the packet at the gateway while taking into consideration the other packets received simultaneously.

The results show a major improvement in terms of QoS and energy consumption however, it is expected that the number of devices will increase over time and will exceed the maximum capacity that can be currently supported by a gateway which will, in turn, increase the rate of loss of packets in different network slice. This challenge motivated us to look for an answer and find a solution to meet scalability challenges and guarantee QoS and energy efficiency for LoRa devices.

Despite the improvement achieved in the results obtained using previous contributions and knowing that more than 30 billion IoT devices are estimated to be connected in future generation IoT networks by 2022, scalability remained an important challenge for next generation IoT networks. In a LoRa network slicing scenario, the current centralized architecture of the state of the art will not be able to handle the challenges of network resource management coming ahead in large scale future IoT deployments. Therefore, in Chapter IV, we propose a Federated SDN-based (DFQL) architecture that addresses scalability challenges by processing data and decision-making to the multi

LoRa agents (MAQL) (based on reinforcement learning) at the edge of the network. The role of slicing decision making and network configuration is leveraged to the gateway which defines the slicing strategy and the local resources that must be reserved after a federation phase with the other nearby gateways (agents) via the global federated orchestrator. By getting closer to the network edge, combining network slicing with SDN advantages, federation tool, and Reinforcement learning technique improve the reliability of communications in a large scale network due its rapid adaptation to changes in a crowded Industrial IoT environment.

Chapter I

Internet of Things (IoTs), Machine learning, and Low Power Wide Area Networks Backgrounds

Summary

I.1 Introduction

For the last couple of years, operators needed to address various challenges and complexities in deploying IoT communications within legacy networks. Inevitably, the expected increase in the number of IoT devices causes saturation problems and will have a large impact on current wireless communication systems. IoT devices mainly require long battery life, extended coverage, larger capacity to support billions of devices with low device and deployment cost. Driven from these requirements, various technologies appeared as potential solutions for IoT network deployment. These technologies are combined with artificial intelligence (AI) tools to overcome challenges and exploiting them to support this technological outbreak which will be one of the most crucial tasks of modern world. In the recent years, the development of AI led to the emergence of Machine Learning (ML) which has become the key enabler to figure out solutions and learning models in an attempt to enhance the QoS parameters of IoT. The purpose of this chapter is to introduce the most emerging technologies nowadays for low power wide area networks LPWAN and Sigfox. We especially focus on the latest research efforts that optimized IoT communications and resource management using emerging technologies that provides virtualization and network softwarization. Mean-

while, we introduce the recent advancements in machine learning models and categories. After that, we shed light on the forth industrial revolution and the Industrial IoT. Finally, we summarizes the critical issues in IoT based on machine learning.

I.2 Communication Technologies

Using its low-cost access to the airwaves, tech innovators took advantage of the unlicensed spectrum to propose promising technologies to support IoT communications. In this section, the focus will be on SigFox and LoRaWAN (Unlicensed technologies) technical overview and their performance in real IoT scenarios.

I.2.1 Sigfox

Sigfox [117] is an ultra narrow-band (UNB) Differential Binary Phase Shift Keying (DBPSK) modulation technology operating on a very small channel bandwidth i.e, 100 Hz in Europe (on a band between 868 and 868.2 MHz) and 600 Hz in USA (on a band between 902 and 928 MHz). Sigfox uses 192KHz of the publicly available band by sending 3 messages using a random frequency to exchange messages over the air. For every transmission, a Sigfox device randomly uses one of the multiple channels available in a bandwidth with a packet duration that goes up to 2 ms [95]. This small bandwidth usage in Sigfox provides the opportunity to concentrate the energy in a very small channel making it very robust against interference. IoT devices transmit short messages in uplink as well as downlink with a throughput that varies between 100 to 600 bits per second depending on the region.

The architecture of Sigfox network is illustrated in **Figure I.1**. It adopts a star topology where IoT devices transmit their messages to the Sigfox network when the radio signal sent reaches the BS within the range. A message can be received by multiple BSs deployed by Sigfox network operators which detect, demodulate and report the messages to the Sigfox Cloud using point-to-point (P2P) links. The Sigfox cloud then pushes the messages to many customer servers and IT platforms. The time on air of a packet is 6 seconds [124] where 6 messages can be transmitted per hour with a

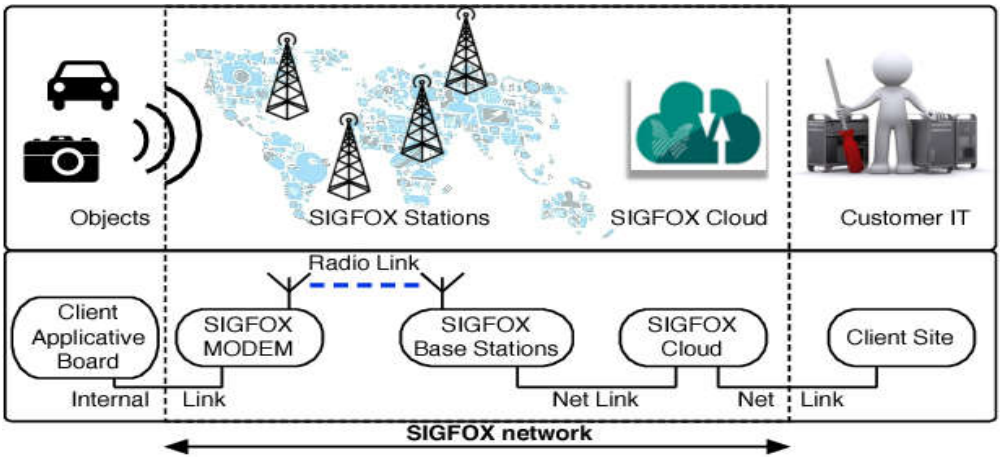


Figure. I.1: SigFox architecture [34]

payload of 4, 8, or 12 bytes. However, in this thesis we are looking towards simulating a larger variety of applications with higher throughput requirements than the one provided by Sigfox. In addition, Sigfox protocol is proprietary which prevented scientists from working on this technology in their research studies. Therefore, we preferred to look towards the possibility of working in LoRa wide area networks (LoRaWAN) for this thesis contributions.

I.2.2 LoRaWAN

LoRa is a shortcut name for **Lo**ng **Ra**nge and a proprietary spread spectrum physical layer that derives from Chirp Spread Spectrum (CSS) modulation as described in the IEEE standard 802.15.4 [51]. CSS modulation transmits symbols by encoding them into multiple signals of increasing or decreasing radio frequencies making signals more robust to multi-path interference, Doppler shifts and fading [15]. Moreover, Forward Error Correction (FEC) and Cyclic Redundancy Check (CRC) techniques are also implemented in LoRa to improve receiver's sensitivity and the robustness of communications. Knowing that LoRa is proprietary and capable of communicating with any other Mediaum Access Control (MAC) layer, LoRa Alliance defines LoRaWAN MAC as an open source protocol built on top of LoRa physical layer. The former defines the communication protocol and system architecture for the network, whereas the latter enables the long-range communication link. LoRaWAN supports low-power and long-range communications where a

set of $I = \{1, 2, ..., i\}$ IoT devices transmit directly to $K = \{1, 2, ..., k\}$ LoRa GWs in a star of stars topology before forwarding data to a backbone infrastructure. LoRa architecture is shown in **Figure I.2** where low throughput traffic is uploaded by each IoT device (thing) to the cloud application servers via IP networks. Within the backbone network, operators servers perform authentication, validation, and forward the packets to the application servers. The latter connects to the backbone network to receive the data and send back the packets in downlink via LoRa GWs.

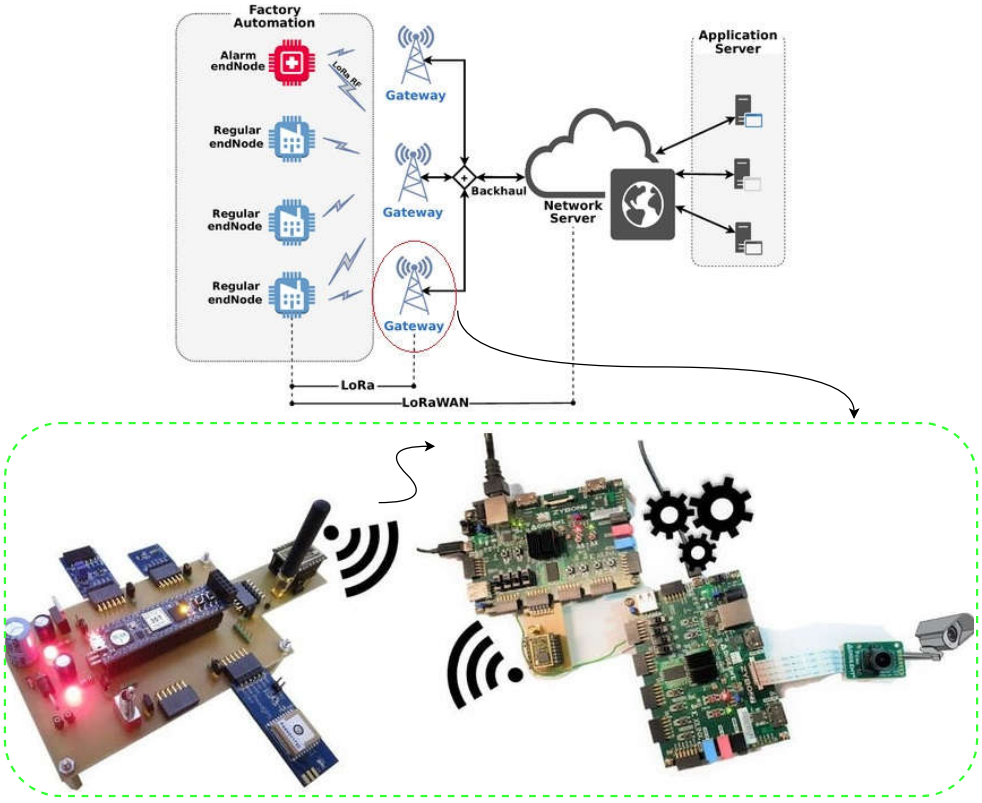


Figure. I.2: LoRaWAN architecture (Industrial applications) [15]

IoT communications are bidirectional where each LoRa device $i \in I$ is characterized with specific parameters that needs to be optimized to meet the requirements of each application in terms of coverage, achieved throughput and energy consumption. In the following we expound LoRa settings and their impact on network performance:

- **Spreading Factor (SF):** SF parameter is by definition the logarithm, in base 2, of the number of chips per symbol and impacts the duration of a Lora chip.

Each device i adopts specific SF configuration for information transmission. LoRa spreads each symbol in a rate of 2^{SF} chips per symbol with $SF = \{7, ..., 12\}$ resulting a data rate computed as written in **Eq. I.1** below:

$$r_{i,c} = SF.\frac{R_{chip}}{2^{SF}} \quad bits/s \tag{I.1}$$

where R_{chip} denotes the chip rate and $r_{i,c}$ the data rate achieved by a device i on channel c of LoRa GW m. Depending on the transceiver model, SF configuration varies from 7 to 12 in a way that higher SF values correspond to more robust communications but lower data rates whereas lower SF values increase the rate and reduce the a active time on air. Some research works ([18] and [17]) claim that SFs are orthogonal to each other, whereas others [31] show that unperfect orthogonality happens between SFs leading to interference between packets. In this thesis, interference in LoRaWAN is considered and will be described in more details later in the interference section below.

- **Transmission Delay:** Transmission delay parameter $d_{i,c}$ denotes the transmission delay of a packet with a length of L bits uploaded by device i to one of the channels c that belongs to GW k.

$$d_{i,c} = \frac{L}{r_{i,c}} \quad seconds \tag{I.2}$$

- **Transmission Power (TP):** Transmission power parameter defines the energy consumed by an IoT device and can be set between -4 and 20 dBm with a step of 1 dB, however in LoRa configuration, TP values vary between 2 and 14 dBm.

- **Carrier Frequency (CaF):** Three different radio bands are available for LoRaWAN (137-175 MHz, 410-525 MHz and 820-1020 MHz). In this thesis, we work on European frequency bands where operators work in in the 863-870 MHz frequency band. Here, specific duty cycles are imposed on IoT devices by the European frequency regulations where each device transmits on a certain frequency in a way respected by both GWs and devices. LoRaWAN channels have a duty-cycle as low as 1% which means that during the last 3600 seconds, a device must never

have transmitted more than 36 seconds in total.

Table I.1: List of parameters

Spreading Factor	Sensitivity (dBm)
SF7	-130.0
SF8	-132.5
SF9	-135.0
SF10	-137.5
SF11	-140.0
SF12	-142.5

- **Radio Bandwidth (BW):** Based on the transceiver model, operators may choose one of the 10 available bandwidth values that varies from 7.8 kHz to 500 kHz. European frequency regulations imposes that the bandwidth adopted for each channel is 125 kHz. Increasing this bandwidth improves the data rate of LoRa device on the expanse of sensitivity. Moreover, increasing the SF value configured on IoT device also reduces the transmitted data rate, increases the strength of the signal and offers a better sensitivity at the GW receiver as shown in **Table I.1**.

- **Co-SF and inter-SF Interference:** LoRa GWs are unable to decode two packets if both are received on the same channel with the same SF configuration. This mechanism leads to packet loss of both packets due to **co-SF** interference. On the other hand, **inter-SF** collisions happen between two packets if they were simultaneously received on the same channel with different SFs and are shown to cause packet loss [31]. Signal to Interference Noise Ratio (SINR) varies based on SF configuration on each device. The assumptions in [86] are followed where a packet should survive interference that comes from other LoRa transmissions. Each device configured with $SF = i$ experiences a SINR value computed based on **Eq. I.3**:

$$SINR_{i,j} = \frac{P_i^{rx}}{\sigma^2 + \sum_{n \in \partial_j} P_n^{rx}} \tag{I.3}$$

where P_i^{rx} is the power of the packet i under consideration sent by device with $SF = i$, σ^2 the lognormal shadowing component and P_n^{rx} the power of one interfering packet $n \in \partial_j$ configured with $SF = j$. Each element in the **Table I.2** [49] denotes the minimum signal power margin threshold $V_{i,j}$, with $i, j \in \{7, ..., 12\}$,

that a packet sent with $SF = i$ must have in order to be decoded successfully over every interfering packet with $SF = j$. Hence, packet survives interference with all interfering packets if, considering all combinations of SF, a higher power margin value (dB) is satisfied than the corresponding co-channel rejection value. One thing to note is that values in below matrix are not symmetric because the power needed to decode a packet is higher when a packet intercepts another one configured with smaller SF. This is because the smaller SF configuration, the stronger the signal power. Taking for example SF8 and SF10, a higher power is needed to decode packets if a packet configured with SF8 intercepts another configured with SF10 (30 dB). However, if the opposite case happened, a smaller power margin value (22 dB) will be needed to decode the SF8 packet intercepted by SF10 packet.

Table I.2: Cochannel rejection (dB) for all combinations of spreading factor for the desired and interferer packets

Desired Packet \ Interferer Packet	SF_7	SF_8	SF_9	SF_{10}	SF_{11}	SF_{12}
SF_7	-6	16	18	19	19	20
SF_8	24	-6	20	22	22	22
SF_9	27	27	-6	23	25	25
SF_{10}	30	30	30	-6	26	28
SF_{11}	33	33	33	33	-6	29
SF_{12}	36	36	36	36	36	-6

- **Propagation Loss model:** The log-distance propagation loss model is adopted to evaluate the performance of LoRa devices in a dense environment and is expressed following to the **Eq. I.4** below:

$$L = L_0 + 10 \cdot \Gamma \cdot \log_{10}\left(\frac{d}{d_0}\right) \tag{I.4}$$

where L denotes the path Loss (dB), d the length of the path in meters (m), Γ represents the path loss distance exponent, d_0 the reference distance in meters (m) and L_0 the path loss at reference distance (dB).

- **Coding Rate (CR):** CR is computed based on **Eq. I.5** in which the redundancy

of the error correction (EC) code is determined and varies between 1 and 4.

$$CR = \frac{4}{4 + EC} \quad with \quad EC = 1, 2, 3, 4 \tag{I.5}$$

- **Adaptive Data Rate (ADR):** ADR is a mechanism for optimizing throughput, energy consumption and time on air (TOA) in LoRaWAN and is generally more efficient for static devices having stable radio frequency (RF) conditions. Depending on the conditions of the environment between the IoT device and the GW, network sever will determine SF and TP values to work on between one of the combinations shown in **Table III.2** below. ADR is highly efficient and very effective in LoRa parameters configuration to maximize battery lifetime, range and overall network capacity. LoRa network server can manage the achieved throughput and the output transmission power used for the communication for each LoRa device individually. The better the coverage the lower the SF and TP configuration. ADR computes the median SNR value of the last 10 received uplink packets, compares it against the SNR limit for each SF and decides afterwards the best configuration.

Table I.3: ADR parameters configurations

Spreading Factor	Transmission Power (dBm)
SF 7	TP 2
SF 8	TP 5
SF 9	TP 8
SF 10	TP 11
SF 11	TP 14
SF 12	TP 14

Multiple research works in the literature evaluated LoRa networks performance [132] [79] [123]. Other research studies focused on evaluating LoRa scalability [90] while considering co-SF interference that comes from collisions when using the same SF configuration on the same channel [47] whereas others assumed that SFs on a channel are perfectly orthogonal [18] [17]. SF represents the ratio between the chirp rate and the data symbol rate and affects directly the data rate and the range that a LoRa device can reach away from a LoRaWAN GW. Moreover, co-SF interference directly impacts

communication reliability, reduces the packet delivery ratio (PDR) successfully decoded at the GW [32] and limits the scalability of a LoRa network when increasing the number of devices [129]. Therefore, scalability should be considered in any upcoming study related to SF configuration strategies and network deployments. Some study examples focused on finding the optimal transmitter parameter settings that satisfy performance requirements using a developed link probing regime [16]. In [83], the authors analyze several SF configuration strategies where a group of LoRa devices can be configured with similar or heterogeneous SFs based on their position from the GW. The goal is to find the scheme that gives the best PDR. However, the impact of SF and TP configuration on network slicing has not been previously tested by the research community.

I.3 Towards enabling programmability in IoT networks

In traditional IoT networks, each equipment requires to be configured separately. This makes maintaining, configuring and adapting network devices to the changes that happen in the device, an expensive and time consuming task [14]. To tackle this problem, Software Defined Networking (SDN) emerged as a promising solution towards enabling programmability, flexibility and virtualization. Nowadays, including various IoT use cases in a single network is not straightforward due to their heterogeneous QoS requirements. Hence, it is hard for operators to guarantee QoS requirements of each service. Network slicing (NS) provides for each use case isolated network resources based on its specific needs. This section defines both SDN and NS paradigms and explores research works that integrates virtualization in IoT networks.

I.3.1 Software Defined Networking

SDN is an approach for network management that enables programmability and decouples the data plane from the control plane without one restricting the growth of other. In a network that requires fast adaptation due to the increasing number of connected devices, managing these elements becomes complex especially in IoT where each device may install various IoT applications and settings. To counter this problem, SDN

emerged as new paradigm [107] that brings the ability to dynamically control the network programmatically through software applications [54].

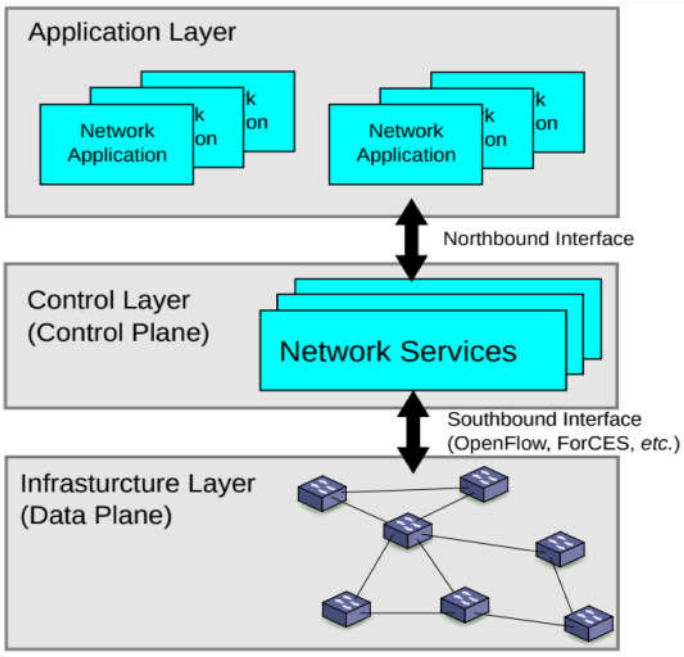


Figure. I.3: SDN architecture [20]

The network architecture illustrated in **Figure I.3** shows how SDN decouples the control plane from the data plane [14] by moving the control logic into the SDN control layer. The control plane programmatically control network resources through a logically abstracted view of the network and expose this view to the application plane where operators configure IoT applications through northbound application programming interfaces (API). This facilitates the task for operators to monitor each softwarized network and to define the forwarding rules based on the traffic and the requirements of devices. OpenFlow [88], is a multi-vendor protocol which defines the interface between the control plane and SDN switches to instruct each switch on how to handle incoming data packets through the southbound API. Hence, the data plane of the network will be only responsible for monitoring local information, gathering statistic and forwarding the traffic according to rules received from the centralized controllers.

I.3.2 Network Slicing

Softwarized and virtualized networks enabled the ability to support heterogeneous services running on top of the same physical infrastructure with each having isolated slice created and managed in an "on demand" manner. Network slicing is an E2E concept covering all network layers and segments. This means that slicing, performed on access, core and transport networks, will provide specific hardware requirements (bandwidth, radio resources, processing power, storage, etc.) across multiple operators [52].

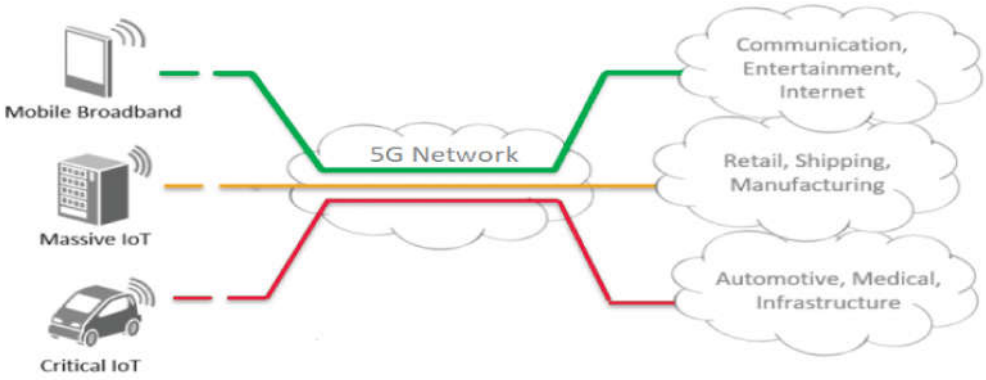


Figure. I.4: Network Slicing architecture

By isolating virtual resources with network slicing, various use cases illustrated in **Figure I.4**, can be served with specific QoS requirements in terms of urgency, throughput and reliability, in a way that removes the impact that may come from a slice over another. However, managing each slice and finding the appropriate amount of resources that should be allocated to each slice, remains an important challenge due to physical resources limitation and the various amount of services required in IoT scenarios. If the requirements for those virtual networks were properly instantiated on physical network infrastructure through orchestrated SDN and carefully designed, network may consume more resources than anticipated, becomes slower, unreliable and impacts other network slices performance. To tackle this challenge, multiple solutions were proposed by the research community to optimize network management in IoT networks and will be listed in detail in the following section.

I.3.3 Network Slicing and SDN integration in IoT

In large scale IoT networks, the cloud-based server should be able to acknowledge more messages as the number of IoT devices in the network increases. Hence, network flexibility is required and potentially reached using network slicing and SDN to provide heterogeneous QoS requirements through isolated E2E virtual networks controlled with SDN to facilitate the task for operators to manage IoT networks. The latter is composed of multi-networks supporting applications with various QoS requirements in terms of reliable delivery and minimum delay [139]. Therefore, authors proposed in [101] a multi-layered IoT architecture involving SDN that is able to cope with various identified IoT challenges, i.e. designing a system able to cope with numerous use cases, ensuring QoS for IoT, controlling congestion and avoiding side effects on legacy services. Moreover, various research works highlighted the efficiency of SDN in IoT networks in terms of security [42], improving transmission quality [119] and scalability through cloud-based solutions [121]. In [127], authors proposed a novel IoT network slicing creation system based on SDN and NFV emerging technologies which provides management flexibility in a centralized fashion. However, all previous solutions are not effective enough to be deployed in upcoming IoT challenges. Therefore, new slicing strategies should be adopted to cope with the fast changes in a more congested IoT environment and to create network slices and allocate physical resources accordingly.

I.3.4 Defining IoT Virtual Slices

In all contributions of this thesis, the first challenge was to propose a classification of IoT devices based on which each service will have isolated and virtualized network resources. Based on the IoT QoS requirements [7] [41], one can note that IoT devices can be classified into three categories proposed in **Table IV.1** below:

Table I.4: Key QoS Requirements of IIoT Network Slices

Slice Name	Packet Delay Budget (ms)	Reliability	Packet Size	Priority	Applications
UCLE	50	1-10^{-4}	24B	1	Emergency action and safeguarding systems
HCLE	100	1-10^{-4}	512B	2	Scale reading applications
LCLE	500	1-10^{-6}	250B	3	Delay tolerant, metering

Ultra high Critical of Latency and Efficiency (UCLE) slice: requires the highest slicing priority due to urgency and reliability requirements of its members. Some examples of these applications are: surveillance and alarm monitoring. Based on **Eq. I.6**, U_{UCLE} is computed to define the utility for critical communications with w_{ld} and w_r the weights of load and reliability, $x_r = SINR_{i,j,k}/SINR_{max}$ the rate of reliability of SINR that a device i achieves on a flow $f_{i,j,k}$ over the highest flow reliability that can be achieved through slice j.

$$U_{UCLE} = x_r(w_r\vartheta_r),\ x_r \in \{0,1\} \tag{I.6}$$

High Critical of Latency and Efficiency (HCLE) slice: requires lower priority consideration and are less critical in terms of delay. This slice presents a trade-off between reliability and load, i.e: health sensors and home security systems.

$$U_{HCLE} = \vartheta_r w_r + \vartheta_{ld} w_{ld} \tag{I.7}$$

Low Critical of Latency and Efficiency (LCLE) slice: requires the lowest priority due to their non-guaranteed data rate and delay-tolerant QoS requirements, i.e: smart metering applications.

$$U_{LCLE} = \vartheta_{ld} w_{ld} \tag{I.8}$$

I.4 The Era of Change: IoT and Machine Learning Trends in Industry for 2020

The Internet of Things (IoT) continues to solidify its position as one of the defining technologies of our time and 2020 could prove its most auspicious year yet. That's because it promises to be the year when 5G goes mainstream, with all the implications that for IoT systems that rely on superior connectivity. But there will be more than 5G to define the upcoming year for the IoT. Many trends likely to characterize the IoT's development throughout the year. Among these trends we note the sustainability will lead the conversation and provide help in several contexts as well as the consciousness of the growing environmental crisis, the merging of IoT, 5G, and Artificial Intelligent

(AI) technologies will give rise to digital manufacturing, especially the fourth IoT industrial revolution (IIoT 4.0), and product hyper-customization will follow, Healthcare will continue to lead the pack, according to the prosper of the "medical IoT", machine learning-based predictive maintenance will energize the IIoT, Edge computing's growth will make security an acute issue, however, Blockchain-based AI and IoT will offer the solution, etc.

In this section we provide an overview of the existing machine learning categories and their applicability in the new generation of IIoT 4.0.

I.4.1 Machine learning categories

The learning activity is essential for the human beings in order to understand and recognize various parameters such as a voice, a person, an object, and others. One generally distinguishes the learning which consists of memorizing information [12] [114], and the learning by generalization [131] [102] in which we usually build a model from learning examples to recognize new examples and scenarios. For the machines, it is easy to handle a large amount of data but difficult to build a good model which is able to effectively recognize new objects in a new test. ML is an attempt to understand and reproduce this learning facility in an artificial system. It therefore seems appropriate to use techniques from this field to discover and model knowledge and reduce the semantic gap [92]. ML is at the crossroads of various fields such as artificial intelligence, statistics, cognitive science, probability theory, optimization, signal and information, and so on [21] [113] [43]. It is therefore very difficult to give taxonomy of machine learning categories. After giving some concepts (section II), we briefly present in this section the four main types of machine learning techniques [33]: Supervised Learning [73], Unsupervised Learning [58], Semi-supervised Learning [29], and Reinforcement Learning [65]. **Table I.5** summarizes the most basic concepts of each ML category and clarifies the differences between them. For more details on categories, readers are invited to read the subsections below.

I.4.2 Future IoT Industry: IIoT 4.0

Due to successive technological advancements, developments, and innovations, the global industrial landscape has drastically transformed over the last years. The Industry 4.0 aims at transforming traditional industries into intelligent ones by incorporating innovative technologies. It enhances and upgrades the current manufacturing facilities, manages systems and technologies to an intelligent level by utilizing key technologies such as IoT, Internet of Services (IoT), cyber-physical systems (CPSs), autonomous, flexible and cooperative robotics, simulations that leverage real-time data and mirror real world into a virtual model, big data analytics, augmented reality (AR), additive manufacturing, information and communication technologies (ICT) and advanced networking technologies. Therefore, industry 4.0 enables physical assets to be integrated into intertwined digital and physical processes thus creating smart factories and intelligent manufacturing environments.

Table I.5: Difference between ML categories

IoT is a rapidly growing technology that has drastically contributed to the Industry 4.0 realization. IoT pursues to pervade our everyday environment and its objects, linking the physical to the digital world and allowing people and "things" to be connected anytime, anywhere, with anything and anyone ideally using any network and service. IoT is regarded as a dynamic and global network of interconnected "things" uniquely addressable, based on standard and interoperable communication protocols and with self-configuring capabilities. Despite still being at the early development, adoption, and implementation stage, Industry 4.0 and IoT can provide a multitude of contemporary solutions, applications, and services. Hence, they can improve life quality and yield significant personal, professional, and economic opportunities and benefits in the near future.

In this context, IIoT is defined as a specific category of IoT which focuses on its applications and use cases in modern industries and intelligent manufacturing. IIoT, which is used in the context of Industry 4.0, can be considered to be a complex system of diverse systems and devices. Through the use of appropriate services, networking technologies, applications, sensors, software, middleware, and storage systems, IIoT provides solu-

ML categories	Input/Output	Purpose	Advantages	Drawbacks
Supervised Learning	Labeled Data/Known Output	• Learn parameters for making predictions	ü More accuracy ü Ability to determine the classes number	■ More computation time in training phase ■ Does not takes place in real time
Unsupervised Learning	Unlabeled Data/Unknown Output	• Illustrate the distribution of data without discriminating between the observed variables and the variables to be predicted	ü Less complexity ü Takes place in real time	■ Less accuracy ■ Analysis results cannot be ascertained
Semi-supervised Learning	Few labeled data+ More unlabeled data/ Few Known Output	• Learn parameters for making predictions • Illustrate the distribution of data without discriminating between the observed variables and the variables to be predicted	ü Does not require a large labeled data set ü High level of accuracy	■ Labelled data is hard to get ■ More computation time in training phase
Reinforcement Learning	Rewards/Actions	• Learning focus on experiences driven sequential decision-making by using rewards where feedback is actions	ü No human intervention ü High level of accuracy	■ More computation time in training phase

tions and functions which develop insight and improve the potential and capability of monitoring and controlling enterprise processes and assets.

I.4.3 Machine learning-based IoT scenarios

Initially, scientists described ML techniques as a tool to generate predictive models for IoT systems. However, wide applications proved ML as a rich domain, which should be understood by those who want to apply it to IoT to get maximum benefits. Application of ML techniques IoT aims to solve numerous issues and offers huge advantages in terms of flexibility and precision.

The designing a network needs to consider various vital issues like topological changes, communication link failures, memory constraints of sensor nodes, computational capabilities, and decentralized management. In fact, ML methods have been successfully adopted to solve several challenges in IoTs scenario such as, localization, Clustering and data aggregation, Event disclosure and Query processing, real-time routing, Medium Access Control, Data Integrity and Fault Detection. **Table I.6** summarizes in a general way, the different IoT challenges, which are solved using ML techniques.

Table I.6: Research work using ML to solve some IoT's challenges

IoT Challenges	**Used ML Techniques**
Quality of Service (QoS)	[130] [85] [133]
Security and privacy requirement	[116] [104] [13] [125]
Interoperability and heterogeneity	[39] [136] [135]
Network congestion and Overload	[57][108] [142]
Network Mobility and Coverage	[71] [134] [97] [28]

I.5 Open issue and future works in machine learning-based IoT

In this section we highlights some IoT's challenges needs to be concerned in the future 5G and IIoT.

I.5.1 Lightweight machine learning approaches for IoT

The large scale deployment of IoTs especially in smart cities environment generate large amount of data. The present machine learning schemes are unable to cope with large amount of dynamic data in real time environment hence much data is wasted without information extraction [2]. The large amount of unlabeled data can be mixed with small amount of labeled data for better convergence of machine learning schemes. In this context, lightweight machine learning approaches can be developed which are suitable to handle large amount of data generated by IoT devices [10]. The concept of data analytics can be used in this regard where sensor location, type and data can help to develop the lightweight models [93].

I.5.2 Distributed machine learning for IoT

The machine learning applications for IoT have to cope with large amount of data. The real data sets for industrial machine learning applications can be thousands of GBs [105]. In such a scenario, the machine learning models which are normally complex and power intensive cannot be run on a single machine. The overall workload can be divided using distribute machine learning with worker machines but it also opens certain challenges to be met [87] [61]. Bandwidth is one of the crucial issues to be faced by powerful worker machines. The worker machines have to frequent exchange data between them at a high transfer rate but such high bandwidth is usually not available which creates a bottleneck [64]. The machines should also need to synchronize them to perform sequential tasks [53]. In realistic scenario, all worker machines are not exactly of identical processing power which slows down the learning and optimization process.

I.5.3 Federated machine learning for IoT

The IoT machine learning applications have to overcome the traditional centralized learning networks that face an increasing challenges in term of privacy preservation, communication overheads, and scalability. In such scenario, the machine learning models which are complex and numerous with heterogeneous gathered data cannot be run in a centralized manner and meet users' QoS [68]. Federated learning networks have been proposed as a promising alternative paradigm to support the training of machine learning models [82]. In contrast to the centralized data storage and processing in centralized learning, federated learning exploits a number of edge devices to store data and perform training distributively [75]. By the way, the edge devices in federated learning networks can keep training data locally, which preserves privacy and reduces communication overheads. However, since the model training within federated learning networks relies on all the edge devices' contributions, the training process can be disrupted if some of the edge devices upload incorrect or falsified training results [69].

I.5.4 Machine learning at the edge for IoT

The huge amount of connected devices has switched the whole network community in a new era called Internet of Things (IoTs) [50]. The concept of IoT has facilitated the community at one end but the delay sensitive and context aware applications have put certain challenges on the performance of lightweight IoT devices [126]. To meet the demand of real time data computing, edge computing has provided promising solutions by executing the data computing requests of IoT devices by some nearby devices [137]. The conventional machine learning may become confuse with the data generated by edge devices due to the fact that it is more complex to identify the real data from complex and noisy environment [63]. Deep learning can play its role in edge devices for better learning and also for keeping the privacy of data preserved during intermediate data transmission [78].

I.5.5 Privacy and security concerns in machine learning approaches for IoT

Machine learning approaches for IoT can be attacked by malicious data which breaches the trust of IoT users [30]. The user data privacy is a fundamental concern of any machine learning scheme which should be taken care of while classification and machine learning [11]. The intrusion detection or malicious data detection can be done with the help of machine learning but they should be light weight to be applicable for IoT based applications [8].

I.5.6 Data Munging for IoT machine Learning

The data collected from IoT devices, which are massive, heterogeneous, inconsistent, and riddled with typos, cannot be used as input for sophisticated machine learning applications [46]. To overcome this issue and get the trends of the data by making it uniform, data gathered from IoTs has to go through a cleansing process [45]. The process is also called as data munging, which commonly includes data exploration, transformation, enrichment exploiting metadata, cleaning or scrubbing the data, i.e. inputting the missing values, removing the unnecessary or invalid data which are not required for getting the

underlying trends of data, and then data validation [91]. This conventional method has several limitations, especially when huge loads of generating data daily from IoT Industry 4.0 and smart cities. For those, the need for accurate and trustworthy analytics in real-time remains crucial, in order to immediately cope with sudden problems, and occurred issues [62].

I.5.7 Adaptive data rate transmission for IoT

IoT devices are generally operated with limited energy batteries hence power consumption is an issue for these devices [115] [96]. To conserve energy, low power consuming protocols to transmit data are famous in IoT. Recently, low power wide area network (LPWAN) has gain attention of researchers to provide low bit rate communication between IoT devices at a long range [4]. Although high data rate infrastructures are also present for IoT e.g. Wi-Fi but they consume large power when connected for a long time. However, depending upon the user requirement, the adaptive scheme can be adopted [23]. Machine learning schemes can be used in this regard to learn the data pattern to decide the data rate requirement for IoT devices [74] [9] [22].

I.6 Problem statement and contributions

After presenting LPWAN technologies summarized in **Table I.7**, we have chosen to work in this thesis on LoRaWAN because its a more scalable technology operating in unlicensed spectrum. Unlike cellular IoT where only hundreds of IoT devices can be simulated in a single cell, Lora is able to serve thousands of IoT devices while also being an alliance with an open approach (instead of the proprietary one SigFox). However, in the state of the art, there's an obvious lack in providing QoS in IoT communications, which till now is limited to just reliability, meaning it's limited to just guaranteeing the delivery of a packet to the base station without considering throughput and delay constraints of the running application. Since the number of connected IoT devices is rapidly growing in the 5G era, an efficient solution to guarantee QoS is by bringing virtualization to IoT networks using SDN and network slicing. The motivation behind it

is to improve server level from end to end across multiple network layers. This guarantees QoS requirements for IoT devices running urgent and reliable applications. We mainly answer the following questions:

- How to assign IoT devices to virtual slices and how to classify these slices?
- How to reserve physical resources for each slice and inside each slice, how to efficiently allocate each device to the appropriate channel ?
- What are the parameters that impact QoS of each device and how to optimize this configuration in a way that doesn't increase network complexity and without impacting network performance ?
- Is current architectures (centralized) capable of supporting IoT communications in large scale IoT deployments and how to efficiently meet the upcoming challenges?

Table I.7: LPWAN technologies comparison for IoT communications

Features	LTE Cat-1	LTE-M	NB-IOT	SIGFOX	LORAWAN
Spectrum	Licensed	Licensed	Licensed	Unlicensed	Unlicensed
Modulation	OFDMA	OFDMA	OFDMA	UNB/GFSK/BPSK	CSS
Rx Bandwidth	20 MHz	1.4 MHz	200 KHz	100 Hz	125-500 KHz
Data Rate	10Mbps	200Kbps-1Mbps	20Kbps	100bps	290bps-50Kbps
Max nb of Msgs/day	Unlimited	Unlimited	Unlimited	140 msgs/day	Unlimited
Max Output Power	23-46 dBm	20 dBm	20 dBm	20 dBm	20 dBm
Link Budget	130 dB	146 dB	150 dB	151 dB	154 dB
Power Efficiency	Low	Medium	Medium High	Very High	Very High
Interference Immunity	Medium	Medium	Low	Low	Very High
Coexistence	Yes	Yes	No	No	Yes
Security	Yes	Yes	Yes	No	Yes
Mobility/localization	Mobility	Mobility	Limited Mobility, No localization	Limited Mobility, No localization	Yes

I.7 Conclusion

LPWAN technologies are being more deployed nowadays in IoT networks due to their efficiency in meeting QoS and energy constraints. However, this proliferation of IoT technologies poses co-existence challenges as they differ in their settings where ones operate in licensed frequency spectrum and others have the ability to communicate via free frequencies spectrum. In this chapter, we presented LPWAN technologies specifications and we listed the latest research work that evaluates their performance and optimization

efforts in improving IoT communications. Moreover, the recent the vital roles of IoT in the recent industry 4.0 are surveyed. Finally, the deployed machine learning model and their issues are highlighted. In the next chapter, we answer the first two questions by first proposing a new methods for assigning IoT devices to the three virtual slices that we have previously defined for IoT communications. Next, we implement network slicing where virtual networks share the same physical infrastructure and we evaluate their performance over different configurations. We show the impact of traditional IoT networks on energy consumption and how the proposed new dynamic slicing and resource allocation strategy contributes in efficiently allocate resources and prioritizing urgent communications over delay tolerant IoT applications.

Chapter II

Online GMM Clustering and Mini-Batch Gradient Descent-based Optimization for Industrial IoT 4.0

Summary

II.1 Introduction

In Chapter II, the problem of providing QoS and flexible resource management for IoT communications is clearly stated. More specifically, three main issues should be tackled towards achieving this goal in Industrial IoT (IIoT) networks:

- Finding the best way to assign IoT devices to the appropriate virtual slice that meets their specific QoS requirements, represents the first challenge in this contribution.
- Finding the best way to reserve inter-slice resources, is the second challenge in this contribution. However, due to capacity constraints and the limited number of channels on LoRa GWs, it is not straightforward to decide on how the amount of resources should be reserved while avoiding resource starvation for any of LoRa virtual slices.
- Defining the best strategy, inside each slice, and allocate intra-slice channels accordingly.

These three issues are directly related in a way that inter-slice resource reservation and intra-slice resource allocation impact not only reliability and QoS, but also the energy consumption of IIoT devices. Great research works recently tackled network slicing in IoT and focused on machine critical communications over various wireless networks. The work in [98] introduced a slicing infrastructure for 5G mobile networking and summarized research efforts to enable E2E NS between 5G use cases. Furthermore, authors in [44] and [111] adopted NS in LTE mobile wireless networks. The former proposed a dynamic resource reservation for M2M communications whereas the latter presents a slice optimizer component with a common objective in both papers to improve QoS in terms of delay and link reliability. In a 5G wearable network, the authors took advantage of slicing technology to enhance the network resource sharing and energy-efficient utilization [55]. Moreover in [37], the authors perform slicing in virtual wireless sensor networks to improve lease management of physical resources with multiple concurrent application providers. In [66], authors proposed several slicing methods for URLLC

scenarios which require strong latency and reliability guarantees. Authors in [77] have proposed an industrial network based on SDN mechanism in order to support dynamic production processes. Unlike traditional industrial networks, remarkable energy saving is achieved. A resource allocation method with consideration of interference management was proposed in [140], where different QoS requirements were guaranteed by optimizing jointly power and sub-channel allocation. Authors in [138] have discussed a 5G network architecture scheme based on SDN to allocate physical resources to virtual slices within a local area and to perform scheduling among slices. An end-to-end network slicing methodology was proposed by authors of [80] in order to share horizontally physical resources whose main purpose is to create multiple virtual networks that can support industry applications. Nowadays, guaranteeing service requirements in many technologies such as LoRaWAN with traffic slicing remains as open research issues [1]. Our main contributions with respect to the surveyed literature are stated as follows:

1. Network slicing is implemented based on LoRaWAN (as a communication protocol) where virtual slices are created and each IIoT devices are assigned to one slice, using a Online Gaussian Mixture Model Clustering (OGMMC) technique, that meet its QoS requirements.

2. A dynamic inter-slicing algorithm is proposed where the bandwidth will be similarly reserved on all gateways based on adopted Mini-Batch Gradient Descent (MBGD). Then the latter is improved and compared to a straightforward static strategy.

3. An energy model is integrated in NS3 simulator to analyze the energy consumed in each slice and an intra-slicing algorithm (Max-Utility) is adopted that meets the QoS requirements of each IIoT device on each slice in an isolated manner.

The remainder of this chapter is organized as follows. Section II.2 presents the proposed network slicing architecture, the system model and the problem formulation established in this paper. In Section II.3, the proposed slicing framework is proposed and well clarified. After the proposed approach is implemented over NS3 simulator. The

framework performance evaluation and simulation results are analyzed and carried out through various scenarios in Section II.4. Finally, Section II.5 concludes this chapter.

II.2 Proposed Architecture and Problem Formulation

In industrial LoRa networks, the general control plane and resource management module are centralized and moved to a management and control entity (MCE) in the cloud to ensure an efficient coordination of resources. Hence, industrial LoRa servers will be the final decision maker in assigning the devices to the appropriate slice and defining the gateway that will transmit the packet. This section will introduce the proposed slicing-based architecture then, the multi-objective problem formulation.

II.2.1 Proposed Network Slicing Architecture

The 5G network architecture design should be built on deep consideration of hardware infrastructure, software control, and the inter-connectivity between them. The network slicing, which can satisfy multiple service requirements based on the unified physical infrastructure and sharing the same physical resources, is considered as a critical challenge by providing multiple instances that operate independently for specific network functions.

The network slicing-based 5G system architecture is given in Figure II.1. The aim of this architecture is to support the creation, control, and management of multiple network slices over factory infrastructures in order to provide high levels of flexibility and scalability and meet QoS requirements for robots, sensors, and actuators in industrial networks. The infrastructure layer contains all the physical resources needed to perform virtualized industrial process. We emphasize that the involved resources go beyond traditional data centers. It includes industrial equipment with sensing and actuation capabilities in addition to the physical computing, storage and network components. The virtualization layer includes the tools and the technologies required to provide a virtualization environment for hosting (VNF) instances. While the slicing layer refers to the deployed industrial slices in order to accommodate specific industrial machinery's QoS.

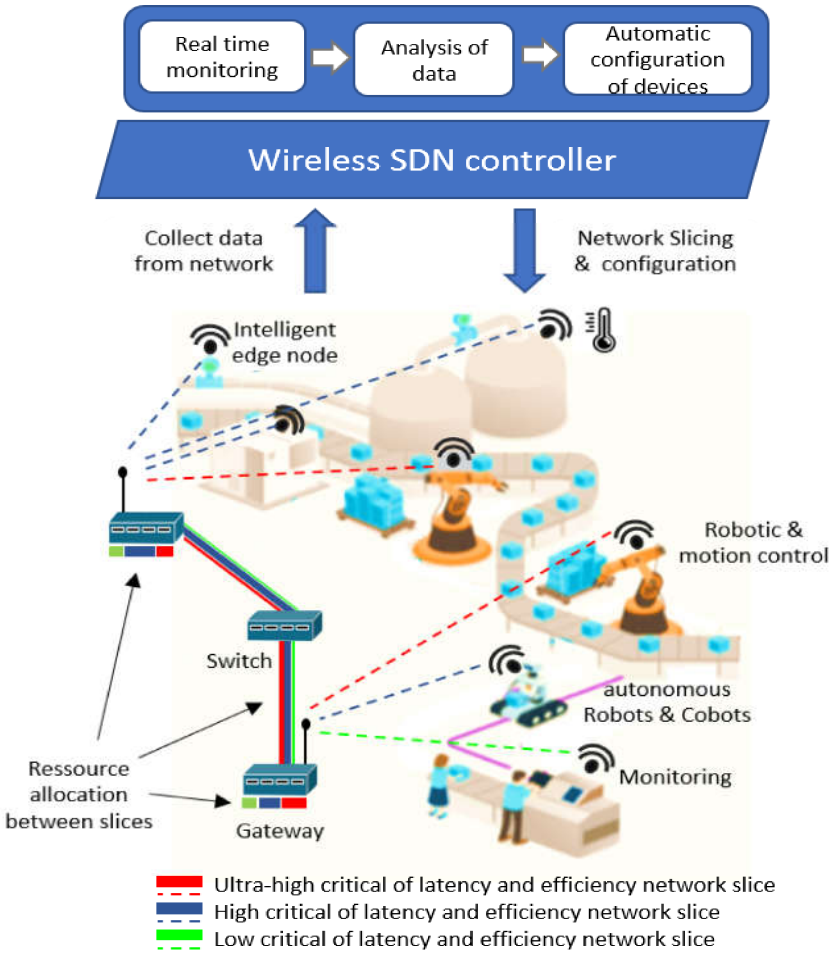


Figure. II.1: Network Slicing Architecture for Industry 4.0

At this level, we define a set of $L = \{l_j, ..., l_L\}$ slices based on throughput R (with bandwidth λ), transmission delay D, and urgency factor ∂. Each slice is responsible for serving a set of $N = \{n_i, ..., n_N\}$ assigned IoT devices through a set of $G = \{g_k, ..., g_G\}$ gateways. Where $\theta_{i,j,k}$ denote the binary value that represent the assignment success of devices i to the slice j through gateway k. All these layers interact with the general administrative control unit represented in the SDN controller, which can control industrial network in a centralized fashion. By interacting with the NFV MANO (Management and Orchestration), SDN controller is in charge of guaranteeing several industrial QoS constraints. However, SDN is defined as a network concept that enables centralized and intelligent control and management that can on-demand modify traffic flows according to industrial application requirements by decoupling data and control traffic. While

NFV is the technology that allows the virtual representation and deployment of logical network functions as Virtual Network Functions (VNFs). When these technologies associated with cloud computing, it can provide better solutions to support reliability and latency requirements, which still represents a major challenge of industrial 4.0 applications. Therefore, it can acquire and allocate virtual resources for slices and enable industrial network reconfigurability, by exploiting the generated virtual flows from devices, in order to meet dynamically industrial devices QoS changes. Assume that the assigned device i to the slice j generate a virtual flow $f_{i,j,k}$ that goes from the gateway k to the SDN controller and is characterized by a utility metrics $U_{i,j,k}$. SDN controller must define also the slice's request, compute the available resources, and serve slices requirements, in a way that avoids resources starvation. At this level, we denote by $C_{j,k}$ the requested physical resources for the slice j on gateway k, while ς_k is the total gateway capacity.

In this work, we aim to jointly optimize industrial machinery QoS and network energy efficiency by providing dynamically slice members with the requested physical resources.

II.2.2 Multi-Objective Optimization Model Formulation

The purpose of this work is to optimize the performance of the industrial network in term of QoS and energy consumption. Toward this goal, the network slicing optimization concept consists of two main steps. Firstly, is to find the best inter-slicing resource reservation strategy. This will impose challenges in aggregating necessary parameters, re-configuring slices, and updating reserved resources. Secondly, is to establish the best intra-slicing resource allocation strategy. This will bring other challenges in reconfiguring devices and updating resource allocation. Before these two phases, IIoT devices need to be assigned to the slice that meets their QoS requirements. These all problems are formulated into multi-objective functions as follows.

II.2.2.1 Quality of Service (QoS) Model for IoT

We define the QoS type in each slice based on the Data Rate R and the Transmission Delay D. The Data Rate model defined in II.1 is denote by the ratio between the peak

rate χ and the slice members n_j. The peak rate of each IoT device depends on many factors such as the received power P_r, the transmit power P_t, and the received SINR, which are written in II.2, II.3, and II.4 respectively [5] [6].

$$R_{i,j,k} = \frac{\chi}{n_j} \tag{II.1}$$

$$P_{r_{i,j}} = \frac{P_t}{L_0} d^{-\delta} h \tag{II.2}$$

$$P_{t_{i,j}} = \frac{SINR(P_I + P_N)}{L} \tag{II.3}$$

$$SINRi, j = \frac{P_r}{P_i + P_N} \tag{II.4}$$

$R_{i,j,k}$ represents the Data Rate for an industrial IoT device i assigned to the slice j on gateway k. Where P_I and L are respectively the receiver noise and the path loss. L_0 is a constant, in which depends on the antenna gains and transmission frequency. h denotes the random variable that represents the channel fading. Finally, d and δ denotes respectively the distance between the trans-receiver and the path loss exponent.

Moreover, the Transmission Delay $D_{i,j,k}$ of a device i assigned to the slice j, is formulated as in II.5. Where $M_{i,j,k}$ is the packet length that is trans-received from a device i.

$$Di, j, k = \frac{M_{i,j,k}}{R_{i,j,k}} \tag{II.5}$$

Based on what was previously mentioned, the QoS cost is modeled by II.6.

$$QoSi, j, k = \overline{R_{i,j,k}} + (1 - \overline{D_{i,j,k}}) \longrightarrow max \sum_{i \in N} QoS_{i,j,k}, \forall j \in L, k \in G, \tag{II.6}$$

where $QoS_{i,j,k}$ is the reimbursements that should be maximized at each slice j and on each gateway k. In addition, $\overline{R_{i,j,k}}$ and $\overline{D_{i,j,k}}$ are adopted as a normalized values that denotes the Throughput and the Transmission Delay achieved by industrial connected device respectively.

II.2.2.2 Energy Consumption Model for IoT

The energy consumption model is considered as the required power to trans-received a data packet, which depends on the received power and the transmitted power given in II.2 and II.7 respectively. In addition, the connected device consumes a P_0 power by the communication module. In this context, the active and the sleep mode, as in II.8, are the two energy states of an IoT device that need to be considered [5] [6].

$$P_{t_{i,j}} = \frac{P_{r_{i,j}} L_0}{d^{-\delta} h} \tag{II.7}$$

$$E_{i,j,k} = [\eta P_{t_{i,j}} + P_0] T_{active} + [\eta P_{t_{i,j}} + P_0] T_{sleep} \longrightarrow min \sum_{i \in N} E_{i,j,k}, \forall j \in L, k \in G, \tag{II.8}$$

where η denotes the electric-to-RF power conversion factor. $E_{i,j,k}$ represents the energy consumption for the device assigned to the slice that should be minimized for each slice.

Addition to the energy and QoS factors, the Packet Error Rate (PER), formulated in II.9, is an interesting factor which reflects the efficiency and reliability of the devices in each slice.

$$PER_{i,j,k} = \frac{S_p}{T_p} \times 100 \longrightarrow min \sum_{i \in N} PER_{i,j,k}, \forall j \in L, k \in G, \tag{II.9}$$

where S_p denotes the number of successful packets and T_p is the number of the total packet sent. PER is the other cost function that should be minimized in order to guaranteed efficiency and reliability in each slice.

II.2.2.3 Network Slicing Problem Formulation

In this contribution, network slicing optimization problem consists of three steps. The first one involves the admission and the assignment of devices to the desired slice. The second stage is the dynamic inter-slicing resources reservation. Although the intra-slice resource allocations are the third stage. Firstly, we define slices based on urgency factor, delay, and throughput, and then we look to assign devices to the slice that meets its QoS requirements. It is noteworthy that the urgency factor, the delay, and the throughput are the key clues for defining the device priority. In addition, other factors

must be considered such as PER, reliability, and the huge load rate of devices to the network. Secondly, we look to estimate the needed inter-slice capacity $C_{i,j}$ based on the required throughput. The purpose of the last step is to optimize the intra-slice resources allocation for each slice members. The Multi-Objective Optimization for the slicing and the resources allocation problem is therefore formulated in II.10.

$$min \sum_{i \in N} \theta_{i,j,k} \frac{E_{i,j,k}}{QoS_{i,j,k}}, \forall j \in L, k \in G \tag{II.10}$$

subject to the following constraints:

$$\sum \theta_{i,j,k} R_{i,j,k} < R^{max}_{i,j,k}, \forall j \in L, k \in G \tag{II.11a}$$

$$\lambda_{r_{j,k}} \cap \lambda_{r_{j',k}} = \oslash, \forall j, j' \in L, \forall k \in G, \tag{II.11b}$$

$$0 < P_{t_{j,k}} < P^{max}_{t_{i,j,k}}, \forall i \in N, \forall j \in L, \forall k \in G, \tag{II.11c}$$

$$\theta_{i,j,k} \in \{0, 1\}, \forall i \in N, \forall j \in L, \forall k \in G, \tag{II.11d}$$

where constraint II.11a guarantees that the sum of transferred traffic by a devices i assigned to the slice j should not exceed the maximum allocated data rate capacity. Moreover, without violating the non-interference principle between slices, constraint II.11b ensures a perfect isolation between them. In other words, each slice has its own bandwidth, whether it was reserved on the same gateway or on different gateways. In addition, each device consumes a transmission power to transfer its traffic data, which should not exceed the maximum transmission power; this is provided by constraint II.11c. Furthermore, the binary assignment value of a device to the slice on the gateway k, is assured by constraint II.11d.

II.3 The Proposed Slicing Approach

The proposed industrial network slicing-based resource allocation scheme, consists of three main steps. Firstly, by using the Online Gaussian Mixture Model Clustering (OGMMC), each device is assigned to the slice that meets its QoS requirements. At the end of this step, the mean throughput for each slice will be estimated. Secondly, radio resources will be dynamically reserved for each slice based on the dynamic and

adaptive Mini-Batch Gradient Descent Algorithm (MBGD). Finally, the preserved radio channels for each slice will be dynamically allocated to the slice members based on the Max-Utility Intra-Slice Resource Allocation algorithm.

II.3.1 IoT devices Assignment: Online GMM Clustering Algorithm

Due to the ultra-diversity of industrial 4.0 services, slices are defined based on urgency, energy and efficiency requirements to meet their objectives. Our proposed architecture is composed of three virtual industrial slices. The first slice called "Ultra-high Critical of Latency and Efficiency (UCLE)" which has the most interest slicing priority and gives more importance to the QoS, efficiency, and reliability. This makes it required by several industrial IoT applications for safety such as emergency action and safeguarding systems. Where "High Critical of Latency and Efficiency (HCLE)" slice gives a less prominence to the latency and considers reliability as a first target. This service is required by the scale readings applications. The last slice is the "Low Critical of Latency and Efficiency (LCLE)", which has the lowest slice priority with non-guaranteed QoS and efficiency. TABLE II.1 outlines the key QoS specifications of the slice in terms of latency, reliability, packet size, and priority factor [67] [84] [66].

Table II.1: Key QoS Requirements of IIoT Network Slices

Slice Name	Packet Delay Budget (ms)	Reliability	Packet Size	Priority
UCLE	50	1-10^{-4}	24B	1
HCLE	100	1-10^{-4}	512B	2
LCLE	500	1-10^{-6}	250B	3

After specifying each slice requirements, IoT devices will be assigned to the corresponding slices. In this context, GMM is adopted as a dynamic and online clustering method (OGMMC) to assign devices to the desired slice, by checking its QoS demands and estimating the mean throughput for slices [27]. Considering a set of J mixture multivariate Gaussian Distributions, indexed by the set of parameters $\Theta = \{\alpha_j, \theta_j\}$, where $\theta_j = \{\mu_j, \sum_j\}$ denotes the Gaussian distribution parameters, in which the mean is denoted by μ_j, $\sum_j$ is the covariance, and $\alpha_j \in \{0, 1\}$ is the mixing probabilities. Assume that all devices data point $\{n_n i, ..., n_N\}$ are independently and identically distributed

according to the mixture probability density function $P(n_i|\theta_j)$. Θ is the parameter that will be estimated to clusters using the Maximum Likelihood Estimation (MLE) process, as in II.12. While the log-likelihood is formulated in II.13.

$$L(\Theta|N) = \Pi_{i\in N} P(n_i i|\theta_j) = \Pi_{i\in N}(\sum_{j\in J} \alpha_j P_j(n_i|\mu_j, \sum j) \tag{II.12}$$

$$log(\Theta|N) = \sum_{i\in N} log(\alpha_j P_j(n_i|\mu_j, \sum j)) \tag{II.13}$$

In view of the structural complexity of II.13, the optimal $\widehat{\Theta}$ cannot be obtained by setting the derivatives to zero. Expectation Maximization (EM) process [94], is a powerful method used to maximize the log-likelihood function and find the optimal parameters. The latter updates iteratively the parameters of individual Gaussian distributions. The given data are considered as incomplete data. This allows defining M latent variables $M = \{m_i, ..., m_M\}$ where each m_i indicates which Gaussian component generates the data vector n_i. The new function that should be maximized is formulated in II.14.

$$\Psi(\Theta, \varphi) = \sum_{i\in N}\sum_{j\in J} log(\alpha_j P_j(n_i|\mu_j, \sum j)) - \sum_{i\in N}\sum_{j\in J} \omega_{i,j}\ \ log(\omega_{i,j}), \tag{II.14}$$

where $\varphi = \{\omega_{i,j}\}$ is the assignment of data vectors (devices) to the clusters. In the expectation step, EM process tends to calculate the probability of a device i belongs to the cluster (slice), by maximizing the Ψ function over the assignment φ and by considering the posterior probability $\omega_{i,j}$ formulated in II.15.

$$\omega_{i,j} = P(m_i = j|n_i, \Theta) = \frac{\alpha_j P_j(n_i|\theta_j)}{\sum \alpha_j P_j(n_i|\theta_j)} \tag{II.15}$$

After that, a maximization of Ψ function over the parameter Θ will be in the maximization step. As a maximization result, the parameters will be estimated and updated iteratively until the convergence. However,the mean, mixing probabilities, and covariance will be tuned accordingly, as presented respectively in II.16, II.17, and II.18.

$$\widehat{\mu}_j^{new} = \frac{\sum_{i\in N} \omega_{i,j} n_j}{\sum_{i\in N} \omega_{i,j}} \tag{II.16}$$

$$\widehat{\alpha}_j^{new} = \frac{1}{n}\sum_{i\in N}\omega_{i,j} \tag{II.17}$$

$$\widehat{\sum}_j^{new} = \frac{\sum_{i\in N}\omega_{i,j}(n_i - \widehat{\alpha}_j^{new})(n_i - \widehat{\alpha}_j^{new})^T}{\sum_{i\in N}\omega_{i,j}} \tag{II.18}$$

The optimal slicing strategy for a limited physical capacity, is to virtually reserve resources for each slice based on the mean throughput R_j^T of its members. After the assignment step, OGMMC calculates R_j^T for slices, as in II.19, with respect to the urgency factor

$$R_j^T = \frac{\sum_{i\in N} R_{i,j,k}}{n_{i,j}}, \forall j \in L, \forall k \in G \tag{II.19}$$

However, each new device sends a connection request to the server that will set up the flag H, in which the Online GMMC algorithm, as in **Pseudo-code 1**, will be re-executed.

Pseudo-code 1 Adaptive OGMMC Algorithm

Input : Set of IoT devices N, Set of Clusters J;
Prameters Θ, $H = 1$, Convergence Criterion ε;
Output: The posterior probability $\omega_{i,j}$, the Tuned parameters Θ;
The Mean Throughput for each slice R_j^T

```
1  begin
2    while H=1 do
3      Define clusters with initialized parameters Θ
       while convergence=false do
4        for each cluster j do
5          for each IoT device I do
6            Assign devices to clusters (slices): II.15
             Maximize the log-likelihood function: II.16
             Update the parameters: II.17 II.18 II.19
7          end
8        end
9        if (Ψ − Ψ_{t+1} < ε) then
10         Convergence ⟵ True
11       else
12         Convergence ⟵ False
13       end
14     end
15   end
16   for each slice j do
17     Compute the mean throughput R_j^T: II.19
18   end
19 end
```

II.3.2 Inter-Slicing Resources Reservation: Dynamic MBGD Algorithm

After assigning IoT devices to the slice that meets its QoS requirements and estimating the mean throughput for slices, we seek in this section to reserve dynamically inter-slices channel resources. To reach this goal, MBGD [112] learning algorithm is adopted as a powerful scheme to improve QoS and minimize cost, in II.10, by finding the optimal throughput parameter needed to split channel resources between slices. In the context of Industrial IoT (IIoT), we assume that slices should be ready to support and serve IoT devices. The global idea is to reserve a minimum capacity level $C_{j,k}$ for slices . Then, by checking slices requirements and learning dynamically devices throughput, more radio resources will be reserved for slices. The adopted MBGD scheme consists of two phases; pre-learning phase and learning phase. In the pre-learning phase (line 1 to 7 in the Pseudo-Code 2), MBGD splits physical resources between slices, by reserving a minimum radio channels based on the estimated mean throughput, even there is no assigned devices (line 4 to 6). With respect to the slice urgency factor, it starts by computing the slice rate γ_j based on R_j^T, as in II.20. This is for defining an appropriate and optimal resource distribution strategy and for not exceeding the maximum gateway capacity. Then, the defined level of physical resources will be reserved, as in II.21. At the end of this phase, the process computes the unserved capacity ξ_k on each gateway, as formulated in II.22.

$$\gamma_j = \frac{R_j^T}{\sum_{j \in L} R_j^T}, \forall j \in L, \forall k \in G \tag{II.20}$$

$$C_{j,k} = \gamma_j \times \varsigma_k, \forall j \in L, \forall k \in G \tag{II.21}$$

$$\xi_k = \varsigma_k - \sum_{j \in L} C_{j,k}, \forall j \in L, \forall k \in G \tag{II.22}$$

Noting that the slice capacity $C_{j,k}$ do not exceed the maximum capacity ς_k provided by each gateway. That is to say, that the mean throughput R_j^T should not exceed the sum of requested throughput for all IoT devices. Therefore, formulas II.19 and II.20 must

satisfy constraints II.23b and II.23a respectively.

$$0 < \gamma_j < 1, \tag{II.23a}$$

$$0 < R_j^T < \sum_{j \in L} R_j^T. \tag{II.23b}$$

After defining a minimum capacity level for each slice, MBGD acts as a brain in each gateway. It learns QoS and energy consumption properties in order to find the best configuration parameter that meets slice demands. This can be done by tracking dynamically slices member's throughput requirements and updates its radio resources capacity.

The common selected parameter in the learning phase (line 8 to 17), is $R_{i,j,k}$ that can interact with QoS, energy consumption and PER. However, when the throughput increase, QoS will be maximized. This refer to the capacity that will be increased according to $R_{i.j,k}$ that decrease the transmission delay as denoted in II.5. On the other hand, the activation time T_{active} of the IoT device will decrease because of the higher speed transmission packets in a large bandwidth. The latter has an effect on energy consumption by decreasing the transmission power and give more chance to increase the percentage of successful transmitted packet. This parameter will be tuned and configured online, using the Mean Square Error (MSE) process, based on the other discussed parameters (training set). In this context, let denote by Γ the current observation values and Γ' is the trained output value formulated in II.24, where is the learning rate. The MSE function is defined in II.25. Where δ_j is the Mini-Batch size and $b_{i,j,k} = 1 - \overline{D_{i,j,k}}$.

$$\Gamma' = \frac{E_{i,j,k}}{QoS_{i,j,k}} \tag{II.24}$$

$$MSE = \frac{1}{2\delta_j} \sum_{i \to \delta_j} (\Gamma - \Gamma')^2 = \frac{1}{2\delta_j} \sum_{i \to \delta_j} (\frac{Ei, j, k}{\overline{R_{i,j,k}} + b_{i,j,k}} - \Gamma)^2 \tag{II.25}$$

The adopted MBGD repeatedly iterates through the training set (power, throughput, and delay) and update the parameter $R_{i,j,k}$ according to the gradient error respectively

in II.27 and II.26.

$$\bigtriangledown_R = \frac{-1}{\delta_j} \sum_{i \to \delta_j} \frac{Ei,j,k}{(\overline{R_{i,j,k}} + b_{i,j,k})^2} * (\frac{Ei,j,k}{\overline{R_{i,j,k}} + b_{i,j,k}} - \Gamma) \tag{II.26}$$

$$\overline{R^{new}_{i,j,k}} = \overline{R_{i,j,k}} - \sigma \ \bigtriangledown_R \tag{II.27}$$

Thereafter, MBGD checks resources demands for each slice and updates slice rate and requested capacity, formulated respectively in II.28 and II.29. Then, it updates reserved

Pseudo-code 2 Dynamic and Adaptive MBGD

Input : Set of IoT devices N, Set of Slices L, MBGD datasets; Set of R^T_j, Stop Criterion ε;
Output: Unserved capacity $\xi^{new}_{j,k}$, allocated capacity $C^{new}_{j,k}$, $R^{new}_{i,j,k}$;

```
begin
    Sorts slices in decreasing order based on urgency factor ∂
     for each Gateway k do
        for each slice j do
            if (R_{i,j,k} = 0) then
                Define a minimum R_{i,j,k} = R^{Tmin}_{i,j,k}
        end
        Define slice rate: II.20
         Define and reserve capacity: II.21
         Compute unserved capacity: II.22
         for each Mini-Batch slice j do
            while convergence=false do
                Compute the gradient: II.26
                 Update the throughput: II.27
                 if | ▽_R − ▽^{t+1}_R | < ε then
                    if R̄^{new}_{i,j,k} < Σ R̄^{new}_{i,j,k} & ξ_{j,k} > 0 then
                        Define slice rate: II.28
                         Define and reserve capacity:II.29 and II.30
                         Compute unserved capacity: II.31
                         Convergence ⟵ False
                         i= i+1; //next iteration
                else
                    Convergence ⟵ True
                end
            end
        end
    end
end
```

and unserved capacity presented respectively by formulas II.30 and II.31. As summarized in **Pseudo-code 2**, the learning phase will be repeated (line 8 to 17) until it serves all slices requirements and stops when there are no more resources to serve it or convergence is reached. We note that, notations with ı*new*ȷ means the predicted value by the proposed learning process. γ_j^{new} is denoted as the new slice rate. $C_{j,k}^{add}$ denotes the new requested capacity to be allocated, $C_{j,k}^{new}$ is the total reserved capacity, ξ_k^{new} is the new unserved capacity, while $R_{i,j,k}$ denote the current observation parameter.

$$\gamma_j^{new} = \frac{\overline{R_{i,j,k}^{new}}}{\sum_{j \in L} \overline{R_{i,j,k}^{new}}}, \forall j \in L, \forall k \in G \tag{II.28}$$

$$C_{j,k}^{add} = \gamma_j^{new} * \varsigma_k, \forall j \in L, \forall k \in G \tag{II.29}$$

$$C_{j,k}^{new} = C_{j,k} + C_{j,k}^{add}, \forall j \in L, \forall k \in G \tag{II.30}$$

$$\xi_{j,k}^{new} = \varsigma_k - \sum_{j \in L} C_{j,k}^{new}, \forall j \in L, \forall k \in G \tag{II.31}$$

II.3.3 Intra-Slicing Resources Allocation: Max-Utility algorithm

After inter-slice resources reservation, we seek in this stage to improve and optimize intra-slice resources allocation. The latter is reached by maximizing utility metric $U_{i,j,k}$ for IIoT devices, in each slice and on each gateway. Utility metric is modeled based on reliability weight (w_r) and load weight (w_{ld}). As mentioned previously in TABLE II.1, slices are different in term of QoS. However, utility metrics can be expressed as follow:

$$U_{UCLE} = x_r(w_r \vartheta_r),\ x_r = \frac{SINR_{i,j,k}}{SINR_{max}},\ x_r \in \{0, 1\} \tag{II.32}$$

$$U_{HCLE} = w_r \vartheta_r + w_{ld} \vartheta_{ld} \tag{II.33}$$

$$U_{LCLE} = w_{ld} \vartheta_{ld} \tag{II.34}$$

where the highest required reliability and urgency for UCLE slice, is denoted by II.32. x_r is considered as a minimum threshold guaranteed during search for the highest reliable link. ϑ_{ld} and ϑ_r are respectively the load rate and the reliability rate. The algorithm summarized in Pseudo-code 3 based on the Analytical and Hierarchy Process (AHP),

searches for the efficient and reliable link that gives the highest utility metric and allocate resources accordingly [35]. Formula II.33 represents the trade-off between reliability and load which explains the less critical latency and priority of HCLE slice and the massive number of connected IoT devices. However, increasing the number of devices will decrease the reliability of links due to congestion. It happens sometimes for devices that are more tolerant to delay, the most reliable link may be overloaded due to the increasing number of devices and should not be taken into consideration. In this case, the process goes to find the optimal link that gives a perfect reliability with minimum load. While LCLE slice modeled by II.34, shows the non-guaranteed latency and QoS requirements. Here, the adopted algorithm seeks to find the virtual link without considering reliability.

Pseudo-code 3 Max-Utility Intra-Slice Resource Allocation

Input : N IoT devices Set, L Set Slices, G Gateways Set;
Output: Max-Utility flows allocation for IoT-device;

```
begin
    for each Gateway k do
        Sorts slices in decreasing order based on urgency factor ∂
        Initialize flow utilities to zero
    end
    for each slice j do
        for each IoT devices do
            Find path with the highest utility U_{i,j,k}
            Allocate IoT device i to f_{i,j,k}
        end
    end
end
```

In general case, we consider a set on IoT device denoted as a source node assigned to slice j, uploads traffic through gateways k. The goal is to find the efficient virtual flow $f_{i,j,k}$ that maximize device utility metric $U_{i,j,k}$, as in II.35, in order to allocate efficiently resources.

$$U_{i,j,k} = U'_{i,j,k} + U''_{i,j,k}, \tag{II.35}$$

where $U'_{i,j,k}$ and $U''_{i,j,k}$ are the utilities provided by each gateway, in which depends on reliability and load.

II.4 Simulation and Results Analysis

In this section, we study the performance of the proposed approach and deeply analyze results. The suggested scheme is implemented in NS3 simulator [25]. TABLE.II.2 summarizes simulation parameters.

Table II.2: Simulation parameters

Parameters	Values
Simulation area	$1km^2$
Power consumption Tx/Rx	25mw
Battery capacity	230mAh
Number of Nodes	1000
Number of gateways (GWs)	1
Used protocol	LoRaWAN
Number of channels	8 per GW
Bandwidth	125 kHz
European ISM sub-band	863-870 MHz

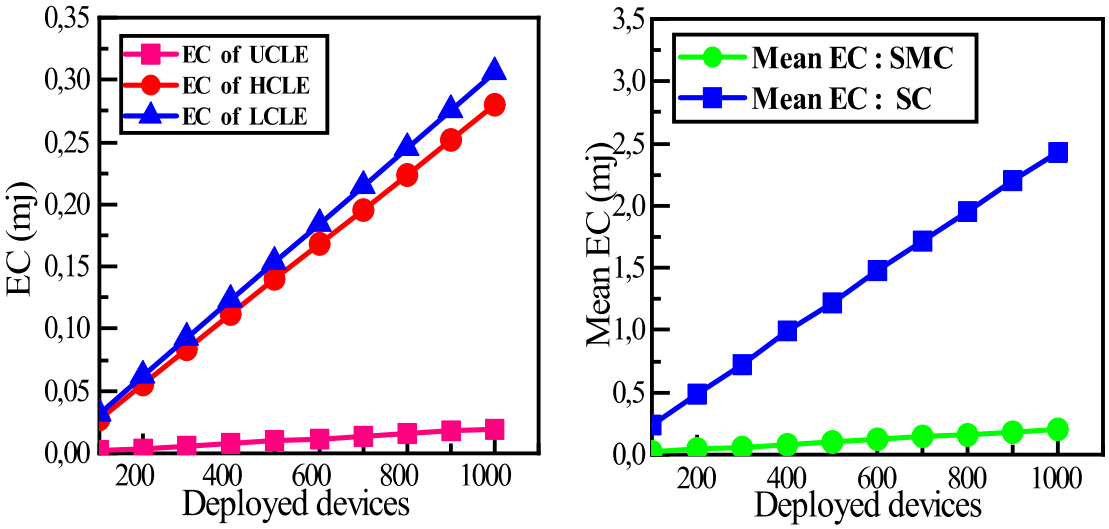
We consider a set of IoT devices initialized with 100 devices, in which increased till it reaches 1000 in a single gateway. They are distributed randomly in an industrial area of one square kilometer. We implement firstly the Slicing Methodology Configuration (**SMC**). Then, Static Configuration (**SC**) will be implemented. The latter has the same simulation parameters to the SMC, but it does not contain the QoS constraint. The objective is to study the QoS profitability for slicing strategy results and make a comparison with the traditional configuration in term of Energy Consumption (**EC**), Transmission Delay (**TD**), and PER.

II.4.1 Energy Consumption Analysis

We assume that the sleep power for IoT devices is set to zero. As presented in Figure II.2a, the total energy consumption for each slice depends on the number of assigned devices and QoS configuration. As in TABLE II.1, LCLE slice is configured with the lowest priority and nonguaranteed QoS with considering load only. As results, a large

number of devices, will be assigned to this slice, in which activation time will be increased proportionally. These leads to increase the power consumption. While HCLE slice scored less power consumption. The latter is configured with a less critical latency but also with a guaranteed reliability and efficiency. As result, a little set of devices will respect this constraint and assigned to HCLE. The tradeoff between reliability and less critical latency leads to minimizing energy consumption compared to the previous slice. While configuration in UCLE slice leads to the most efficient power consumption compared to the others. This return to the inter and intra process that gives higher importance and priority for QoS configuration, and reliability in Utility calculation. It allows the little set of assigned devices to take the most reliable gateway with smaller duration of spectrum occupation time.

For more evaluation, the mean EC for proposed scheme was compared to the EC for the SC. As denoted in Figure II.2b, the EC for static method increases exponentially while number of deployed devices increases. This because that all IoT devices are assigned to the same network without respecting reliability, QoS, and efficiency constraint. This prove the efficiency of the proposed slicing method.

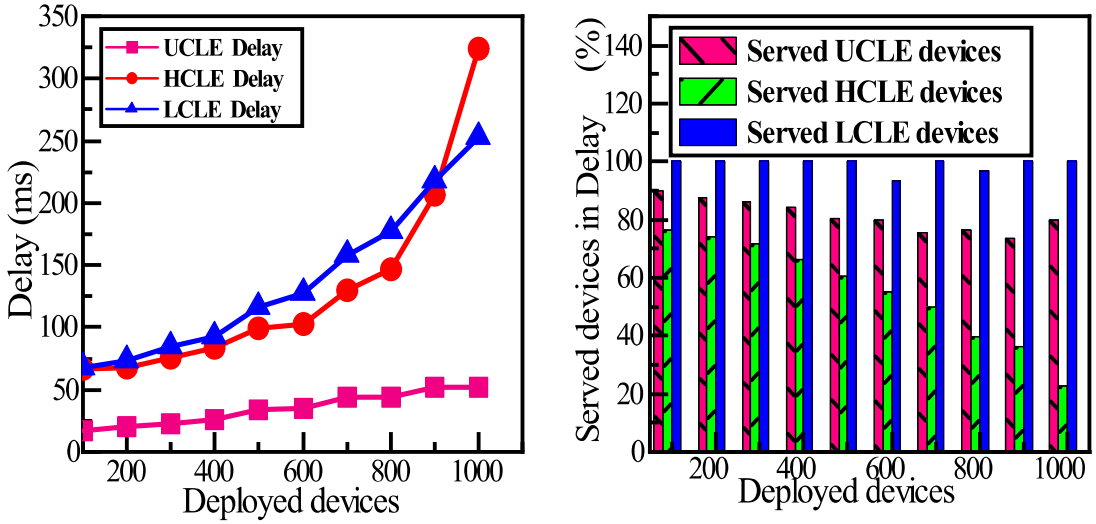


(a) Energy Consumption for slices (b) Mean Energy Consumption: SMC vs. SC

Figure. II.2: Energy Consumption evaluation

II.4.2 Delay Variation Analysis

Relying on QoS class in TABLE II.1, each slice is configured with packet size constraint. It is noteworthy that HCLE has the highest packet size than LCLE and UCLE slices. While UCLE slice is configured with the highest reliability and efficiency. This will give mores chance to UCLE slice to serves its members, by maximizing utility metrics and allocating more radio channels. As results, throughput will be increased, allowing delay to be reduced to a minimum. This increases the percentage of devices that have not violated their delay threshold. This is not the case for HCLE, which is configured with medium priority, large packet size, and a utility depending on reliability and load. As the number of deployed devices increases, delay will increase exponentially, even if the learning process reserves more resources. Thus, increase the percentage of IoT devices that violated its delay threshold. Instead, delay in LCLE will be increased according to the increase of its member but it remains less than HCLE. This refers to the QoS constraint that configure LCLE with a little packet size than HCLE and consider the load only. Figure II.3a demonstrates the delay variation for slices, while Figure II.3b demonstrates the percentage of served devices in delay.



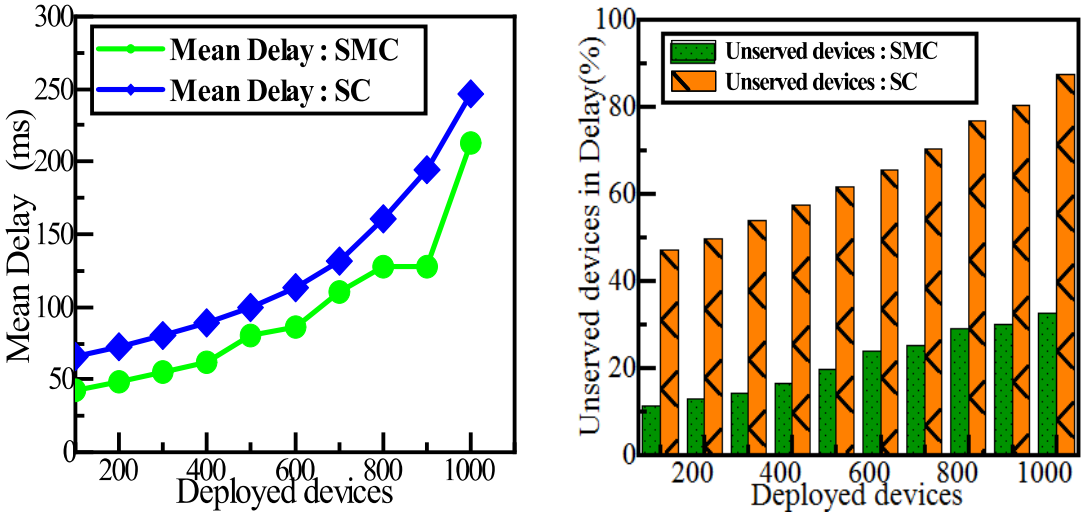
(a) Delay variation of slices

(b) Percentage of Served device in delay: SMC vs. SC

Figure. II.3: Delay variation and Percentage of Served devices

More performances evaluations were conducted, and the proposed slicing method was

compared to the static method in term of delay variation and percentage of unserved devices in delay. As seen in Figure III.5c and II.4b, SC had the worst results with highest delay varies exponentially with the increase of deployed devices number. Also 88% of devices did not respect their delay thresholds on 1000 deployed devices compared to 32% of the SMC. This refers to the random configuration that did not take into consideration QoS requirements of devices.



(a) Mean Delay variation: SMC vs. SC

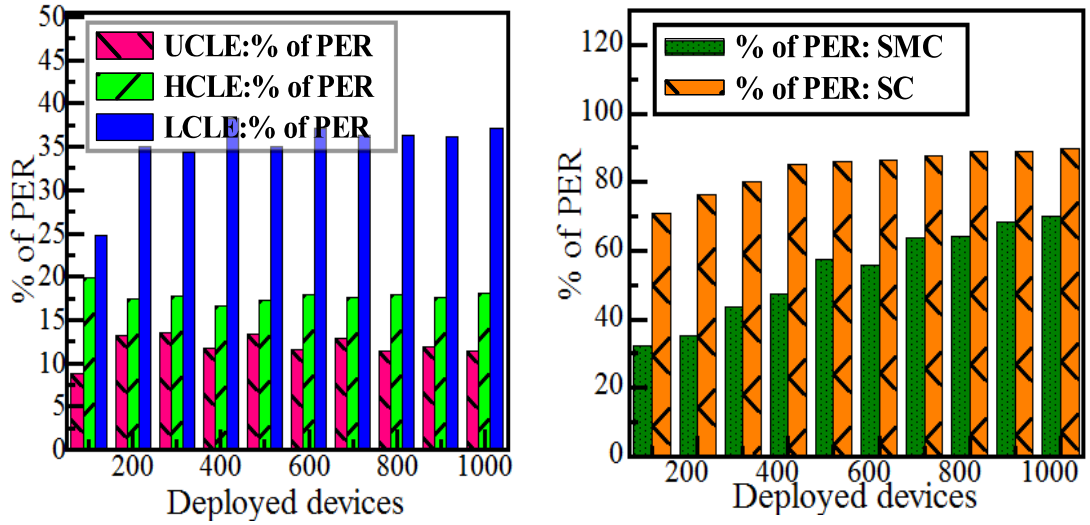
(b) Percentage of Unserved device in delay: SMC vs. SC

Figure. II.4: Mean Delay variation and percentage of Unserved devices

II.4.3 Packet Error Rate Analysis

Figure II.5a, shows the evolution of PER in each slice. As described previously, the proposed process gives more importance to the UCLE slice, by checking its QoS demands and serve it. Then, it moves to the HCLE slice, and later if there are unserved resources it can reserve some to LCLE slice. As results, UCLE slice will be frequently served, and may therefore limit PER of its members. It is remarkable that PER in UCLE increase when devices increase at 200 and 300, then PER decrease at 400 devices, etc. This returns to the learning tools that try to avoid resource starvation in each slice and dynamically reserves channels following throughput demands. As the deployment device increases, congestion increases, and reserved resources will no longer be sufficient to support devices requirements. At this stage, PER will be increase until process serve

its demand in future iteration. In fact, it is the same thing for the other two slices, but with considering less QoS constraint to HCLE and no QoS constraint to the LCLE slice. This implies that fewer radio channels will be reserved, in which congestion will be increased, while PER will also increase. By the other hand, with static configuration, PER was highly increased. This refers to the continuously increased congestion with the increase of deployment devices. This because the QoS configuration is not considered. Results in Figure II.5b prove the efficiency of the optimization process in reducing PER with 40% compared to 20% of the SC.



(a) Percentage of PER of slices (b) Mean percentage of PER: SMC vs. SC

Figure. II.5: Percentage of PER evaluation for both configurations

II.5 Conclusion

The ever-increasing exploitation of smart devices with improved capabilities, is leading to a radical change in the industrial landscape. However, low latency, reliability, and efficiency are required to support new Industrial 4.0 challenges. Network slicing paradigm benefits can be extended to address out-range performance requirements. In this chapter, network slicing is implemented and investigated in centralized standard industrial network architecture in which inter-slice resource reservation and intra-slice resource allocation methods are both proposed and optimized with respect to the QoS requirements of each slice members. Firstly, OGMMC was proposed to assign devices

to the desired slice that meet its QoS demands and to estimate mean throughput slices requirements. Secondly, Mini-Batch based slicing scheme was proposed to reserve dynamically radio channels to slices. Finally, Max-Utility algorithm was adopted in order to efficient allocate the gateway resources to the slice's members. Various evaluation results are provided and analyzed after proving the isolation concept between virtual slices. Although, a comparison is provided between the static and the proposed slicing strategies. However, the dynamic network slicing appears to be the best slicing method compared the static one in this chapter. Simulations results show and prove the efficiency of the proposed framework in saving energy consumption, reducing delay, and PER.

We believe that these results can still be improved if IoT devices' parameters were efficiently optimized to improve network performance in each slice. In the following chapter, instead of considering the dynamic slicing mechanism which jointly increases or decreases both spreading factor and transmission power (LoRa technology) of an IoT device, we propose to deeply evaluate our slicing-based optimization scheme in multiple industrial scenarios exploiting LoRa technology in a way that maximizes network performance of each slice in smart Industrial LoRaWAN scenario.

Chapter III

In-Depth Performance Evaluation of Network Slicing Strategies in Large Scale Industry 4.0

Summary

III.1 Introduction

The solution that we proposed in Chapter II using network slicing, has shown its worthiness in providing urgency and reliability in LoRa networks. In this context, an urgent packet will always have a part of LoRa resources reserved to guarantee its arrival to the gateway. However, after analyzing in depth reliability results, we have noticed that there is still room for improvement to reduce the percentage of packets lost in the network. Hence, we decided to investigate more in depth on how LoRa parameters impact QoS of an IoT device and how to configure the latter properly in a network slicing scenario.

Reliability in LoRa does not depend only on just successfully delivering a packet to a channel above sensitivity, it also depends on the configuration of other packets received at the same time on a LoRa channel which may cause significant packet losses due to co-SF and intra-SF interference. The former happens when two packets configured with same SF are simultaneously received at the same channel whereas the latter happens when the interfere packet is decoded with different SF configuration. Many research studies focused on proposing various SF configurations and distribution strategies over multiple network deployments [100] with the goal to overcome capacity limits [122] and to provide a trade-off solution that minimizes energy consumption while maximizing reliability [76]. However, SF is not the only parameter that should be taken into consideration when optimizing LoRa configuration.

Increasing TP of a device is also important to increase SNR and the chance of decoding one of the packets upon interference. However, one should also not forget on battery constraints that should be respected to avoid depleting the battery lifetime of

IoT devices. In some works, authors showed the importance of configuring IoT devices with a proper combination between SF and TP parameters to improve scalability of LoRaWAN [103] and to avoid performance degradation and unfairness that happens in LoRa network if IoT devices configure SF and TP locally [109]. LoRa originally includes a link-based adaptation of SF and TP configurations using the ADR mechanism. Many works tried to propose modified and improved ADR algorithms with the goal to increase reliability and energy-efficiency without taking into consideration the possibility of intra-SF and inter-SF collisions [70] [118] [110]. The latter can be decreased with the knowledge of the entire network or by finding the optimum configuration after testing all combinations of LoRa parameters that respects specific thresholds [16]. However, this method is considered as time consuming because sometimes, achieving multi-objectives in terms of reliability and energy-efficiency do not always require tuning parameters, especially on IoT devices placed at the edge of their communication range [26]. In [81], the performance of the official ADR mechanism proposed by LoRa is evaluated and shows the impact of different configurable parameters in terms of slow convergence rate which introduces higher energy consumption and packet losses.

All works previously mentioned from the literature improved LoRaWAN performance using various optimization strategies. However, the random-based access nature in IoT network gives the motivation to optimize network slicing with a slice-based parameters configuration that treats each virtual slice differently without considering all IoT devices as devices belonging to the same LoRa network. The goal behind this proposition is to improve QoS of IoT devices and limit interference and collisions in each LoRa virtual network. This chapter contributions extend the previous one by considering smart industrial 4.0 application belonging to different QoS classes and are stated as follows:

1. We include QoS in LoRa, which was previously considered as a best-effort technology, with the goal to test the flexibility that network slicing provides in terms of traffic management and QoS integration.

2. We apply the best slicing strategy found in Chapter II, where the bandwidth is efficiently reserved on each LoRa GW separately based on MBGD prediction, MLE estimation, and Static configuration. The goal of this scheme is to avoid channels starvation while considering dynamically the exact need of each slice starting by the one with the highest slicing priority.

3. We adapt the TOPG, proposed by [36], as a slicing optimization scheme based on Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) and Geometric Mean Method (GMM). The proposed method efficiently configures LoRa SF and TP parameters and improves the performance of each slice in terms of QoS, reliability and energy consumption.

The remainder of this chapter is organized as follows. We devote Section III.2 to describe the network slicing system model in a smart industrial scenario and the multi-objective optimization problem established in this chapter. Section III.3 presents the proposed slicing and optimization algorithms implemented over the LoRa module of NS3 simulator [86]. The performance evaluation of the algorithm and simulation results are analyzed and carried out through various scenarios in Section III.4. Finally, Section III.5 concludes the chapter.

III.2 Network slicing Architecture Modeling and Problem Formulation

III.2.1 Network Slicing in LoRa-based Industrial Network Modeling

In a smart industry network deployed with LoRa, heterogeneous use cases are enabled for connected machines in terms of robotics mobility, supply chain management and optimization, asset tracking and optimization, quality management, smart communication, and high data transfer speed, etc. However, due to the heterogeneity of these applications, a single smart industry network is unable to support all of these traffic types within a network without compromising QoS for any of them. In case of an accident, a connected robots and machines should immediately communicate the information to

the global server responsible for emergency situations. However, this information could be lost or arrived without respecting the required delay in industry area. Hence, the focus here is on applying traffic slicing in smart industrial scenarios, virtually isolated, and with specific QoS thresholds. In **Table III.1**, the key QoS requirements of UCLE, HCLE and LCLE slices were previously defined in Chapter I with each having running various IoT applications illustrated in **Figure IV.1**. One of the listed use cases is smart mobility, where, an increase in communication delay between two machines or between a machine and its gateway may result a dangerous problem and should be provided with the highest levels of urgency and reliability. On the other hand, some smart industry applications only require best-effort behavior like metering and actuating to measure the consumption data of different resources like product parts presence, heating power, etc.

The major challenge in IIoT communications is to support various applications having heterogeneous QoS requirements in terms of latency, reliability, packet size, and slicing priority. The key QoS requirements of IIoT virtual slices in the new 5G generation are summarized in Table III.1.

Table III.1: Key QoS Requirements of IIoT Network Slices

Slice Name	Packet Delay Budget (ms)	Reliability	Packet Size	Priority
UCLE	50	1-10^{-4}	24B	1
HCLE	100	1-10^{-4}	512B	2
LCLE	500	1-10^{-6}	250B	3

We consider an industrial network slicing architecture based on SDN, LoRa technology, and virtual IIoT slices, as illustrated in **Figure. IV.1**. The first slice called "Ultra-high Critical of Latency and Efficiency" (UCLE) is characterized with the highest slicing priority due to its most critical urgency and the highest degree of importance given to the QoS, efficiency, and reliability. Some UCLE applications are required to ensure safety, such as emergency action and safeguarding systems. The second slice is denoted as "High Critical of Latency and Efficiency" (HCLE) which gives less prominence to latency but still considers reliability and efficiency in IIoT communications. An example of HCLE applications are scale readings. The last slice is named as "Low Critical of Latency and Efficiency" (LCLE), which has the lowest slicing priority and

the highest packet delay budget with non-guaranteed QoS and efficiency [67][84].

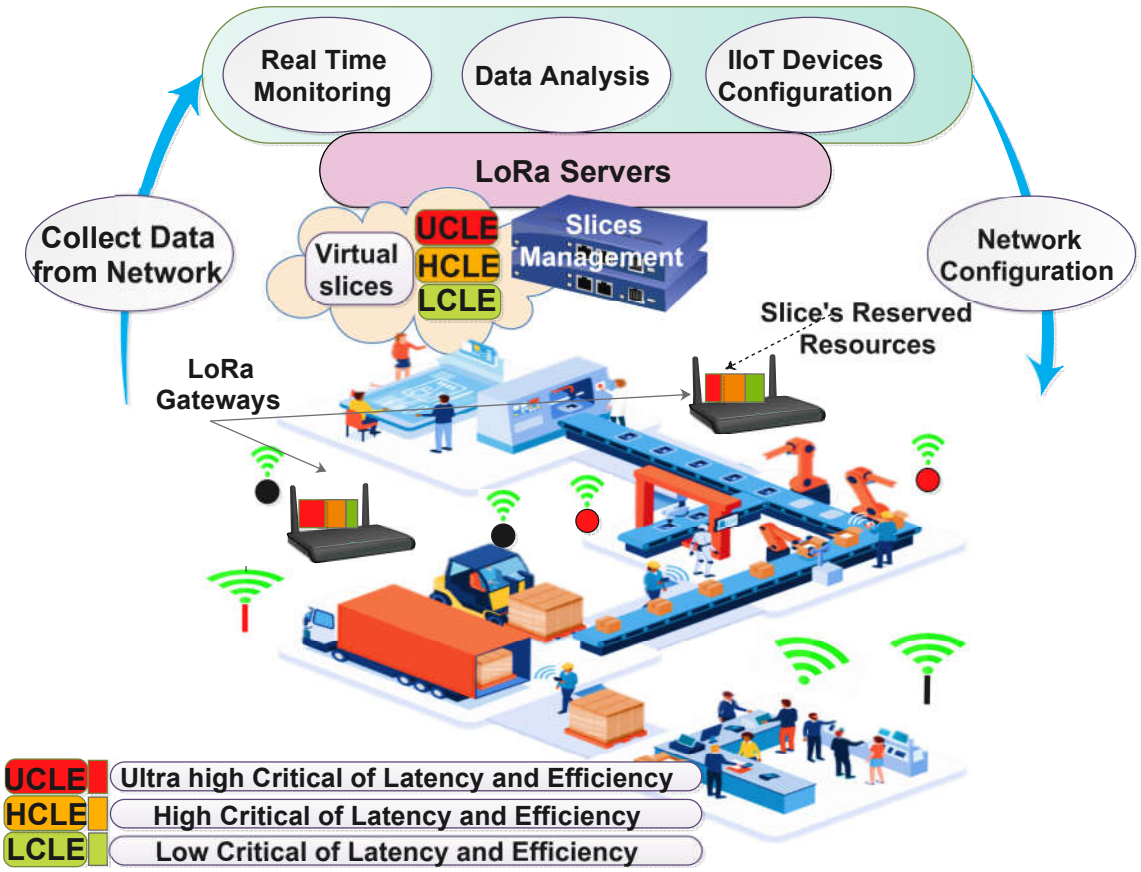


Figure. III.1: Network Slicing-based LoRaWAN Architecture.

Figure. III.2 illustrates how IoT devices are connected to a LoRa GW in the actual standard architecture (**Figure. III.2a**) and configured with one of the SF-TP combinations, listed in **Table III.2**.

Table III.2: ADR parameters configurations

Spreading Factor	Transmission Power (dBm)
SF 7	TP 2
SF 8	TP 5
SF 9	TP 8
SF 10	TP 11
SF 11	TP 14
SF 12	TP 14

While **Adaptive Data Rate (ADR)** is a mechanism for optimizing throughput, energy consumption and time on air (TOA) in LoRaWAN and is generally more efficient for static devices having stable radio frequency (RF) conditions. Depending on the

conditions of the environment between the IoT device and the GW, network sever will determine SF and TP values to work on between one of the combinations shown in **Table III.2** below. However, the server, in our industrial scenario, aims to increase both SF and TP values simultaneously to increase signal robustness and decode packets at larger distance from the GW. Nevertheless, when network slicing is applied on a Lora gateway (**Figure III.2b**), ADR mechanism becomes inefficient specially if the device in question belongs to a slice having specific QoS thresholds that needs to be respected before reaching external LoRa servers through the internet. With traffic slicing, the receipt of urgent communications is now guaranteed at the GW level. However, overestimating SF and TP configurations leads to an increase in energy consumption due to the longer activity time for an IoT device when uploading a packet with high SF configuration. Moreover, if a high SF is configured, achieved throughput may be lower than the one that needs to be guaranteed in the corresponding slice. Hence, for each slice, one should not be limited to discrete SF and TP values proposed by LoRa ADR mechanism. This work enables the possibility to define specific slice-based SF and TP combination to be configured on an IoT device in a way that respects its QoS thresholds.

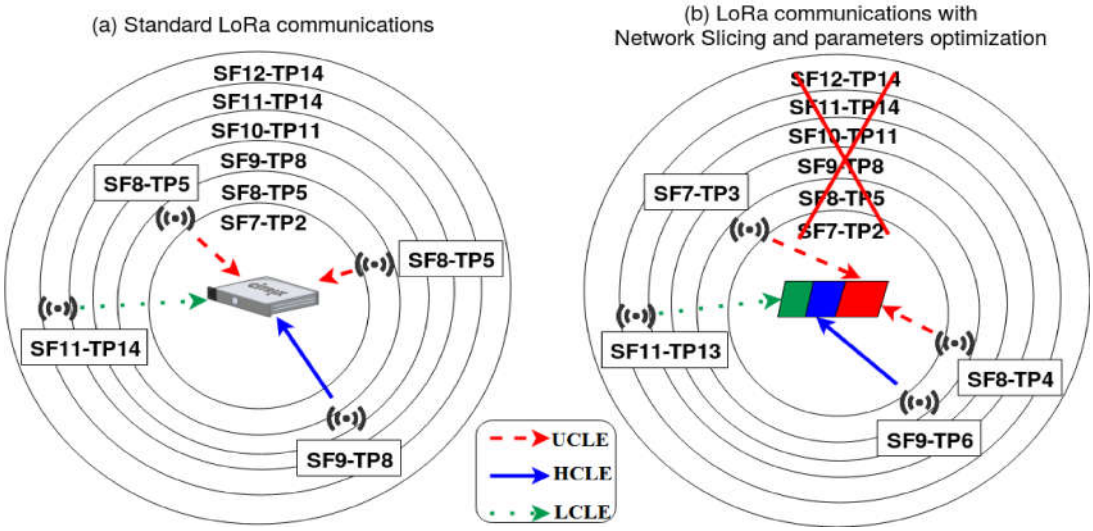


Figure. III.2: (a) Standard LoRa and (b) LoRa network slicing with parameters optimization

In this Smart industrial scenario network, we assume that centralized LoRa servers are aware of the QoS required by each device in terms of delay, throughput and reliability.

Moreover, LoRa servers are responsible for defining resource reservation strategies on LoRa gateways (GWs) and on configuring the devices with SF and TP parameters. It is noteworthy that to improve communications in an IoT environment, multiple objectives should be reached. More precisely, we jointly consider in this chapter QoS, energy, and reliability requirements as major key factors and objectives to optimize parameters configuration of an IoT device belonging to a slice with a specific slicing priority ∂. On each LoRa gateway, a slicing rate is estimated based on the throughput required by the devices active in each slice $l \in L = \{l_j, ..., l_L\}$, based on throughput R (with bandwidth γ_r), transmission delay D, in order to define capacity C_{jk} that needs to be reserved. Each slice is responsible for serving a set of $N = \{n_i, ..., n_N\}$ assigned IIoT devices through a set of $G = \{g_k, ..., g_G\}$ gateways. We denote by C_{jk} the requested physical resources for the slice j on gateway k, while ζ_k being the total gateway capacity. Let Θ_{ijk} be the binary value that represents the assignment success of devices i to the slice j through gateway k. We search to jointly optimize QoS and network slicing energy efficiency by assigning slice members with the proper SF and TP configurations and dynamically providing the requested physical resources to these members. However, solving this multi-objective problem is challenging. Therefore, the goal in this chapter is to optimize parameters selection after evaluating the cost and benefits in each slice.

III.2.2 Problem Formulation

As the main contribution is to improve QoS and energy consumption for IIoT devices in Industry 4.0, we formulate in (III.1) the multi-objective function into an optimization model which should be minimized at each slice and for each device.

$$Min \sum_{i} \Theta_{ijk} \frac{E_{ijk}}{QoS_{ijk}} \ , \forall j \in L, \forall k \in G, \tag{III.1}$$

where $E_{i,j,k}$ represents the energy consumption of devices i assigned to the slice j through gateway k. In addition, the energy consumption depends on other factors like the transmit and the received power, and devices mode (active or sleep), etc. [5] [6]. Whereas, $Q_{i,j,k}$ denotes the quality of service of devices i assigned to the slice j through gateway

k that denote the reimbursements that should be maximized at each slice and for each gateway. The QoS, as in (III.2), is defined in each slice based on the Data Rate R and the transmission delay D, as in (III.3), for each device i assigned to the slice j through gateway k.

$$QoS_{ijk} = \overline{R_{i,j,k}} + (1 - \overline{D_{i,j,k}}), \tag{III.2}$$

$$D_{ijk} = \frac{M_{i,j,k}}{R_{i,j,k}}, \tag{III.3}$$

where $M_{i,j,k}$ is the packet length that is trans-received from a device i. In addition, $\overline{R_{i,j,k}}$ and $\overline{D_{i,j,k}}$ are adopted as a normalized values that respectively denote the throughput and the delay achieved by an IIoT connected device.

Due to the multi-objectivity of the problem, we search to find the optimum slicing strategy with the proper SF and TP configurations that simultaneously maximize QoS benefits of each slice and minimize energy and reliability costs without under optimizing a function over another. This multi-objective problem is formulated subject to the constraints below:

$$\sum_{l \in L} \Theta_{ijk} = 1, \forall k \in G \tag{III.4a}$$

$$\gamma_{j,k} \cap \gamma_{j',k} = \emptyset, \forall j, j' \in L, \forall k \in G \tag{III.4b}$$

$$0 \leq P_{i,j} \leq P_j^{max}, \forall i \in N, \forall j \in L \tag{III.4c}$$

$$\sum_{i \epsilon N} \Theta_{ijk} R_{j,k} \leq R_{j,k}^{max}, \forall j \in L, \forall k \in G \tag{III.4d}$$

$$\Theta_{ijk} \in \{0, 1\}, \forall i \in N, \forall j \in L, \forall k \in G \tag{III.4e}$$

The first constraint (III.4a) ensures that each device should always choose exactly one and only network slice even if the latter was implemented on different physical

gateways. Moreover, a perfect isolation is guaranteed in (III.4b) between two bandwidth parts assigned for two different slices regardless if the latter was reserved on the same or on two different gateways. The transmission power of each device is limited in constraint (III.4c). Furthermore, constraint (III.4d) guarantees the sum of uplink traffic sent by slice members which do not exceed the maximum data rate capacity of the slice that can be sent through each gateway. Constraint (III.4e) ensures binary association values of device k to slice l.

III.3 The proposed Slicing strategies: Dynamic MBGD prediction, MLE estimation, and Static configuration

In this section, we focus on the inter-slicing radio resource reservation based on both schemes MBGD prediction and MLE. While for the intra-slicing resource reservation we adopt the TOPG, proposed by [36], to find the best SF and TP parameters configurations in a way that meets best the requirements of the corresponding slice member. **Fig. III.3** illustrates the proposed slicing mechanism contribution in this chapter. Meanwhile, the IoT devices to the virtual slice admission and assignment is well described in [89] and [36] respectively. Next, the main difference between these techniques and the static strategy will be clarified. Although the adopted TOPG will be explained in this section.

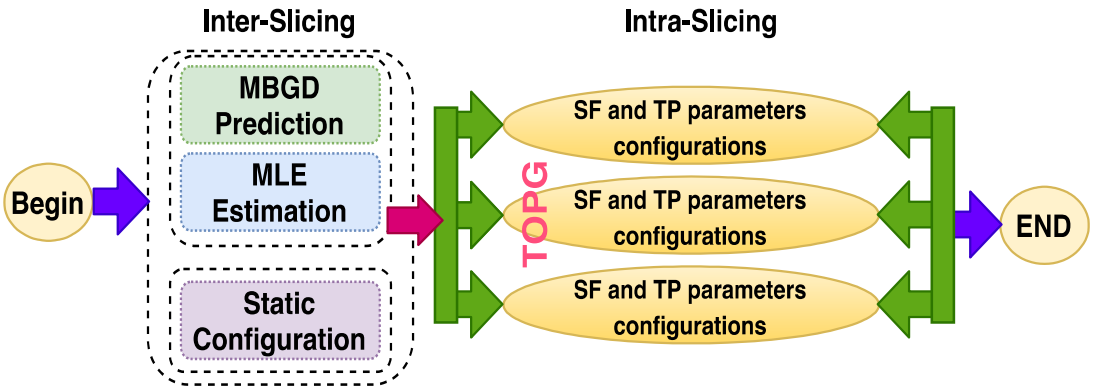


Figure. III.3: The proposed slicing mechanism

III.3.1 Dynamic MBGD prediction-base Inter-slicing

The dynamic MBGD inter-slicing radio resource reservation method is proposed by [89] in Chapter II to reserve radio channels based on MBGD learning algorithm and Mean Square Error Process (MSE) process [112]. MBGD (**Pseudo-code 4**) is considered as an efficient machine learning scheme adopted to find the optimal amount of radio channels, that should be reserved for each IIoT network slice, by predicting the optimal required throughput based on the MSE kernel.

Pseudo-code 4 Dynamic and Adaptive MBGD-based slicing strategy

Input : Set of IoT devices N, Set of Slices L, MBGD datasets; Set of R_j^T, Stop Criterion ε;
Output: Unserved capacity $\xi_{j,k}^{new}$, allocated capacity $C_{j,k}^{new}$, $R_{i,j,k}^{new}$;

begin
  Sorts slices in decreasing order based on urgency factor ∂
  for *each Gateway k* **do**
    for *each slice j* **do**
      if $(R_{i,j,k} = 0)$ **then**
        Define a minimum $R_{i,j,k} = R_{i,j,k}^{Tmin}$
      end
      Define slice rate: II.20
      Define and reserve capacity: II.21
      Compute unserved capacity: II.22
      for *each Mini-Batch slice j* **do**
        while *convergence=false* **do**
          Compute the gradient: II.26
          Update the throughput: II.27
          if $\mid \bigtriangledown_R - \bigtriangledown_R^{t+1} \mid < \varepsilon$ **then**
            if $\overline{R_{i,j,k}^{new}} < \sum \overline{R_{i,j,k}^{new}}$ & $\xi_{j,k} > 0$ **then**
              Define slice rate: II.28
              Define and reserve capacity:II.29 and II.30
              Compute unserved capacity: II.31
              Convergence ⟵ False
              i= i+1; //next iteration
          else
            Convergence ⟵ True
          end
        end
      end
    end
  end
end

In this context, the global idea is to attribute a minimum capacity level for slices in order to make slices ready to serve its members. Then, by investigating slice online requirement, and dynamic QoS learning, MBGD reserves appropriate radio channels number for each slice on each GW. Following this strategy, the proposed technique consists of pre-learning phase and learning phases. In the former, MBGD serves a minimum capacity level to each slice based on the mean throughput even there is no assigned devices to. It starts by computing the defined slice rate, as in (III.5), in which the maximum gateway capacity δ_k has been respected. Then, as denoted in (III.6), the defined resources will be reserved. At the end of this phase, MBGD updates the unserved capacity Υ_k following the formula (III.7).

$$\gamma_j = \frac{R_j^T}{\sum_{j \in L} R_j^T} \tag{III.5}$$

$$C_{j,k} = \gamma_j * \delta_k, \forall j \in L, \forall k \in G \tag{III.6}$$

$$\Upsilon_k = \delta_k - \sum_{j \in L} C_{j,k}, \forall j \in L, \forall k \in G \tag{III.7}$$

MBGD acts as a brain in each gateway, after defining a minimum capacity level for each slice. It defines the best slicing configuration, meeting each virtual slice demands, by checking dynamically slices member's throughput. In this regard, the transmission rate parameter is defined as a dedicated instructional parameter that will affect the QoS and energy consumption when changing its behavior. However, by increasing the throughput, the slice' reserved channels number will be increased while the transmission delay of a transmitted packet will be decreased. Hence, at its turn, the IIoT device activation time will decrease due to the higher transmission rate of packets in a larger bandwidth. Furthermore, the configured transmitted power remains constant, the decrease in delay impacts energy consumption of a device, which will decrease when packets occupy the spectrum for a smaller amount of time without negatively decreasing the percentage of received packets. This parameter will be tuned online, using MSE and MBGD techniques, as illustrated in **Pseudo-code 4**. At this stage, Ω is defined to be the current observation value and $\Omega^{'}$ is the trained output value, as in (III.8). We define the MSE

cost as illustrated in (III.9). Noting that Z_j is the mini-batch size for each slice.

$$\Omega^{'} = \frac{E_{i,j,k}}{QoS_{i,j,k}} \tag{III.8}$$

$$MSE = \frac{1}{2Z_j} \sum_{i \in Z_j} (\frac{E_{i,j,k}}{\overline{R_{i,j,k}} + b_{i,j,k}} - \Omega)^2 \tag{III.9}$$

At each iteration, MBGD has repeatedly set and update, through the training set (power, throughput, and delay), the parameter $R_{i,j,k}$ according to the gradient error, which respectively defined in (III.11) and (III.10). Where υ is the learning rate.

$$\bigtriangledown_R = \frac{-1}{Z_j} \sum_{i \in Z_j} \frac{E_i, j, k}{(\overline{R_{i,j,k}} + b_i)^2} * (\frac{E_i, j, k}{(\overline{R_{i,j,k}} + b_i, i, k)} - \Omega)^2 \tag{III.10}$$

$$\overline{R^{new}_{i,j,k}} = \overline{R_{i,j,k}} - \upsilon \bigtriangledown_R \tag{III.11}$$

Based on the updated throughput, MBGD learns resources demands, for each slice, and updates the slicing rate (γ_j) and the requested capacity ($C^{toadd}_{j,k}$). Then, it computes the new reserved and unserved capacity which respectively denoted by $C^{new}_{j,k}$ and $\Upsilon^{new}_{j,k}$. More details is provided in [89].

III.3.2 MLE Estimation-base Inter-slicing

The dynamic MLE inter-slicing radio resource reservation method is proposed by [36] to estimate and reserve the appropriate resources for slices by finding the maximum likelihood buffer demands for each slice j starting by the one with the highest slicing priority. However, the traffic that needs to be uploaded follows a Poisson distribution and LoRa servers are aware of the amount of data stored in the buffer B_i of each slice member.

Int This context, R_i is denoted as the throughput observed by each device $i, \forall i \in N$ captured at each slicing interval time and identified by a corresponding probability distribution. For a fixed physical capacity, the optimum slicing strategy is to virtually reserve resources for each slice based on the mean throughput of its members. Therefore, R_i follows a Poisson distribution P(λ) where λ denotes the throughput needed by

device i assigned to slice $j, \forall j \in L$. Meanwhile, $f(R_i|\lambda)$ is defined as the probability density function similar to $L(\lambda|R_i)$ that represents the likelihood of λ given the observed throughput.

$$
\begin{aligned}
L(\lambda|R_1, R_2, ..., R_N) &= f(R_1|\lambda)f(R_2|\lambda)....f(R_N|\lambda) \\
L(\lambda|R_1, R_2, ..., R_N) &= \prod_{i=1}^{N} \frac{e^{-\lambda}\lambda^{R_i}}{R_i!} \\
logL(\lambda|R_1, R_2, ..., R_N) &= log\left[\prod_{i=1}^{N} \frac{e^{-\lambda}\lambda^{R_i}}{R_i!}\right] \\
logL(\lambda|R_1, R_2, ..., R_N) &= \sum_{i=1}^{N} log\left[\frac{e^{-\lambda}\lambda^{R_i}}{R_i!}\right] \\
logL(\lambda|R_1, R_2, ..., R_N) &= \sum_{i=1}^{N} \left[log(e^{-\lambda}) + log(\lambda^{R_i}) - log(R_i!)\right] \\
logL(\lambda|R_1, R_2, ..., R_N) &= \sum_{i=1}^{N} \left[-\lambda + R_i log\lambda - log(R_i!)\right]
\end{aligned}
$$

To find the maximum likelihood parameter, we apply the first derivative and solve it to zero.

$$
\begin{aligned}
\frac{\partial logL(\lambda|R_1, R_2, ..., R_N)}{\partial\lambda} &= \sum_{i=1}^{N} \left[-1 + \frac{R_i}{\lambda}\right] \\
&= -N + \frac{\sum_{i=1}^{N} R_i}{\lambda} = 0 \\
\widehat{\lambda} &= \frac{\sum_{i=1}^{N} R_i}{N}, \forall i \in \{1, ..., N\}
\end{aligned}
$$

To prove that the $\widehat{\lambda}$ is the maximum value, we apply a second derivative as follows:

$$
\frac{\partial^2 logL(\lambda|R_1, R_2, ..., R_N)}{\partial_\lambda^2} = -\frac{\sum_{i=1}^{N} R_i}{\lambda^2}, \forall j \in L
$$

The obtained result is always a negative number which indicates that $\widehat{\lambda}$ is maximum and the optimal parameter to consider. Hence, the best slicing decision is to consider the mean throughput $\widehat{\lambda_l}$ of slice j members $\forall j \in L$. However, slices are not equal in terms of priority. Therefore, GW resources will be dynamically allocated to the most urgent slice starting by the channel with the highest reliability. Let $\gamma_j = \widehat{\lambda}_j / \sum_{j=1}^{L} \widehat{\lambda}_l$ be the slicing rate

based on which the algorithm reserves for each slice a capacity $c_{j,k} = \varsigma_k.\gamma_j, \forall j \in L$.

Pseudo-code 5 Dynamic MLE-based Inter-Slicing Strategy

Input : Capacities c_k;
Number of slices j;
Set of Throughput Requirements R_N

```
1  begin
2      Put slices in decreasing order based on priority sp_l
       if method=Dynamic − Sicing(DS) : all_slice then
3          for each GW m do
4              Apply MLE Estimation based on the throughput required by all slice j members
               Define Slicing Rate γ_j and Reserve capacity c_{j,k}
5          end
6      else if method=Adaptive − DS(ADS) : each_slice then
7          for each GW k do
8              for each slice j ∈ L do
9                  Apply MLE Estimation based on the throughput required by slice j members in the range of GW k
                   Define Slicing Rate γ_j and Reserve capacity c_{j,k}
10             end
11         end
12     else
13         Reserve capacity c_{j,k} equally between slices
14     end
15 end
```

Output: Set of resources reserved for each slice l

Pseudo-code 5 summarizes the inter-slicing algorithm and starts with the most critical slice **(line 2)**. Depending on the slicing strategy, the algorithm equally reserves the bandwidth between slices based on a straightforward *"Fixed Slicing"* **(line 14-16)** or estimates the needed throughput $\widehat{\lambda_i}$ of all slice l members in the case of *"Dynamic Slicing"* strategy, defines Θ_l for channels reservation and reserve a part of the bandwidth on all LoRa GWs in a similar manner **(line 3-7)**. If the *"Adaptive Dynamic Slicing"* was adopted, slicing rate of each slice Θ_l varies from a GW to another because in this case, MLE estimates throughput of each slice members deployed in the range of the corresponding GW m **(line 8-14)**. The algorithm moves next to the following slice, repeats the process and stops when no resources are left for reservation.

III.3.3 The Proposed TOPG Optimization Algorithm

After defining slicing objectives, next we need to adapt the weight of every objective before optimizing SF and TP parameters configurations in a way that meets best the requirements of the corresponding slice. To do this, we propose an optimization algorithm based on GMM and TOPSIS methods.

For this step, $A_J{=}(a_{ij,J})_{n\times n}$ denoted as the judgment matrix where $a_{ij,J} > 0$ and $a_{ij,J} \times a_{ji,J} = 1$. Each value $a_{ij,J}$ measures the importance of an objective i over another objective j for each slice J. Based on the importance values in each slice, a priority vector is derived and denoted as $\psi_l = (\psi_{1,J}, \psi_{2,J}, ..., \psi_{(n-1),J}, \psi_{n,J})$, where $\psi_l \geq 0$ and $\sum_{i=1}^{n} \psi_i = 1$, from the decision matrix A_J. With GMM, weight configuration for each objective is defined as an objective function of the following optimization problem:

$$\begin{cases} Minimize \sum_{i=1}^{n} \sum_{j>i} [ln(a_{ij,J}) - (ln(w_{i,J}) - ln(w_{j,J}))]^2 \\ w_{i,J} \geq 0, \sum_{i=1}^{n} w_{i,J} = 1, \end{cases}$$

which have a unique solution and can be simply solved by the geometric means of the rows of each slice's decision matrix A_J:

$$w_{i,j} = \frac{\sqrt[n]{\prod_{j=1}^{n} a_{ij}}}{\sum_{i=1}^{n} (\sqrt[n]{\prod_{j=1}^{n} a_{ij}})} \tag{III.12}$$

After finding the objective weights for each slice, we import the weight vector of each slice into a decision matrix D_J, which consists of a set of possible alternatives A_x as shown in the below matrix:

$$D_J = \begin{array}{c} Alternatives \\ A_1 \\ .. \\ .. \\ A_J \end{array} \begin{array}{c} \begin{array}{cccc} w_{1,J} & .. & w_{n-1,J} & w_{n,J} \end{array} \\ \begin{pmatrix} a_{1,1} & .. & a_{1,n-1} & a_{1,n} \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ a_{m,1} & .. & a_{m,n-1} & a_{m,n} \end{pmatrix} \end{array}$$

where each value $a_{x,y}$ represents a parameter configuration of a device with $y \in \{1, 2, ..., n\}$ defines the objective and $x \in \{1, 2, ..., m\}$ denotes a combination of SF $i \in \{7, ..., 12\}$ and TP discrete values $j \in \{2, ..., 14\}$ in dBm among which LoRa servers need to assign the device with the best configuration based on W_J, the set of objectives weight values of the corresponding slice. TOPSIS method requires normalized values $\overline{a_{x,y}}$ in D_J with the goal is to find the alternative with the shortest distance from positive ideal solution and the one with the largest distance from the negative ideal solution.

$$\overline{a_{x,y}} = \frac{a_{x,y}}{\sqrt{\sum_{x=1}^{m} a_{x,y}^2}}, \quad with \quad x \in \{1, ..., m\}, y \in \{1, ..., n\} \tag{III.13a}$$

In other terms, the goal is to find the best configuration that maximizes QoS benefits and minimizes the costs in terms of PLR and energy consumption. For each positive ideal solution A^+ and negative ideal solution A^-, normalized weight rating $v_{x,y}$ can be determined using the following equations:

$$v_{x,y} = w_{x,J}\overline{a_{x,y}}, \quad with \quad x \in \{1, ..., m\}, y \in \{1, ..., n\} \tag{III.13b}$$

$$A^+ = (v_1^+, v_2^+, ..., v_n^+) \tag{III.13c}$$

$$A^- = (v_1^-, v_2^-, ..., v_n^-) \tag{III.13d}$$

where V_y value results using equations

$$V_y^+ = \left\{ \max_x v_{x,y}, y \in Y_1; \min_x v_{x,y}, y \in Y_2 \right\} \tag{III.13e}$$

$$V_y^- = \left\{ \min_x v_{x,y}, y \in Y_1; \max_x v_{x,y}, y \in Y_2 \right\} \tag{III.13f}$$

where Y_1 and Y_2 respectively respect benefit and cost criteria. We calculate next the euclidean distance from the positive ideal solution and negative ideal solution of each

alternative; respectively as follows:

$$d_i^+ = \sqrt{\sum_{j=1}^{n} (d_{i,j}^+)^2} \tag{III.13g}$$

$$d_i^- = \sqrt{\sum_{j=1}^{n} (d_{i,j}^-)^2} \tag{III.13h}$$

where $d_{x,y}^- = V_y^+ - v_{x,y}$, with $x = 1, ..., m$ and $d_{x,y}^- = V_y^- - v_{x,y}$, with $x = 1, ..., m$.

$$\zeta_x = \frac{d_x^-}{d_x^+ + d_x^-} \tag{III.13i}$$

Finally, the configurations are ranked according to the relative closeness previously calculated, and each device will be assigned with the configuration that provides the highest value ζ_x due to its closest position to the positive ideal solution.

III.3.4 Static vs Slicing Strategies

The proposed scheme based prediction process will be evaluated and compared to the other approach cited before in large scale LoRa in realistic Industry 4.0 scenario. The main difference between these schemes is summarized in **Fig. III.4**.

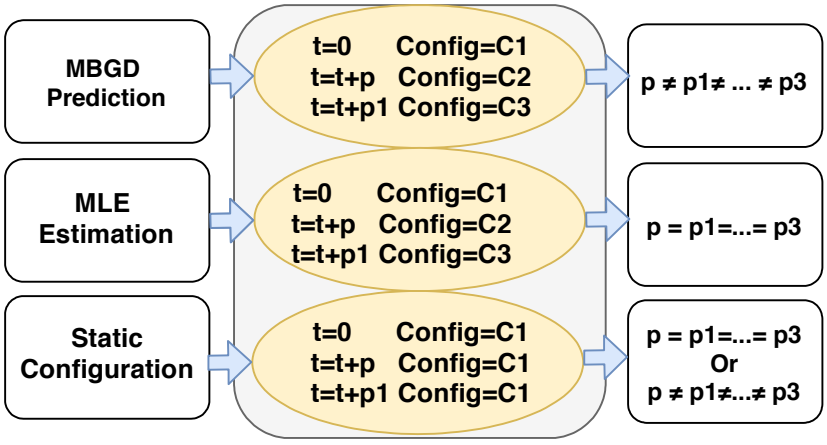


Figure. III.4: Main difference between strategies.

In Static scheme the number of radio channel reserved are fixed statically and manually at the beginning. This means that every t with all the periods p are similar to each other or different, the network keeps the same resource's configuration all the

time. Unlike MLE estimation strategy, the process updates every period p its configuration on the resource reservation knowing that these periods are similar to each other $p-1 = p-2 = .. = p_n$. While MBGD prediction strategy update its configuration every time and every period, which depends directly to the tracking of QoS performance. Thus means that all time periods are not similar to each other $p \neq p_1 \neq p_2 \neq ... \neq p_n$.

Thus, the main algorithm that will be used in our industrial scenario to provide an in-depth evaluation of our scheme, is provided in **Pseudo-code 6**.

Pseudo-code 6 Slicing SF-TP Configuration

Input : GW Capacities; Set of slices L;
Set of Throughput Requirements $R_{i,j,k}$

begin
  Put slices in decreasing order based on priority ∂_j
  for *each GW* k **do**
    for *each slice* $j \in L$ **do**
      if *predict=true* **then**
        Apply **MBGD** based on slice required throughput.
      else
        Apply **MLE** based on slice required throughput.
      end
    end
  end
  for *each GW* k **do**
    for *each slice* $j \in L$ **do**
      Apply GMM to define $W_{l,n}$ of each objective.
    end
  end
  Sort devices in $i_{j,k}$ based on urgency factor ∂_j.
  for *each device* $i \in n_{j,k}$ **do**
    Apply TOPSIS to define (SF-TP) parameters:
    SF_i,TP_i=**TOPSIS**($w_{j,1}, .., w_{j,n}$)
    Configure the device with SF_i and TP_i.
  end
end

Output: Set of resources reserved for each slice j.
(SF-TP) parameters configuration for each device i.

III.4 Simulation Results

In this section, we present the extensive performance comparison between static, estimation-based and prediction-based slicing strategies. For this job we used the open source NS3 simulator due to its strong tracing architecture and the ability to implement realistic industrial IoT scenario. Table III.3 gives a brief of LoRa parameters implemented in this work. We assume that devices are defining a random time for transmission but periodically uploading small packet payloads of 18 Bytes. We consider a large industrial area in which IoT devices and LoRa gateways LoRa devices and gateways are both placed over a cell of 10 KM radius following to a uniform random distribution. Each device is configured with spreading factors that varies from 7 to 12 when uploading traffic to LoRa GWs. Each GW is characterized by 8 receiving channels wih each channel having a bandwidth of 125 kHz in the 867-868 MHz european sub-band.

Simulation Parameters	
Cell Radius	10 KM
LoRa devices and GWs distribution	Random Uniform
Propagation loss model	Log-distance
Bandwidth	125 kHz
Spreading Factor	{7,8,9,10,11,12}
European ISM sub-band	863-870 MHz
Power Consumption Parameters [15] [19]	
Battery Maximum Capacity	950 mAh
LoRa Supply Voltage	3.3V
Amplifier Power's added Efficiency	10%
Connected (Tx/Rx-SF7 to SF12)	1.58 to 25.11 mW
Standby	0.09 mW
Sleep	0 mW

Table III.3: Simulation Prameters

III.4.1 LoRa Configurations Impact

In the first study, we fix the number of devices to 2500 devices and we evaluate the performance of each slicing strategy with multiple LoRa configurations. The spreading

factor (SF) and transmission power (TP) are two main parameters for any transmission to avoid interference and guarantee the receipt of all packets. The following static configurations are tested ($SF7-TP2$, $SF8-TP5$, $SF9-TP8$, $SF10-TP11$, $SF11-TP14$, $SF12-TP14$) as well as $RAND$ configuration in which the spreading factor and the transmission power are configured randomly and finally ADR which is the adaptive data rate configuration currently adopted by LoRa. Following to the **Table III.4** illustrates PLR percentage for each category in each slice, it is noteworthy that packets lost when the gateway is saturated due to the load in the network, due to co-channel rejection or to lack of sensitivity when the packet is out of range or it does not reach the gateway due to an appropriate SF configuration. It is remarkable that the ADR is the best configuration which will be used for the rest of the simulation.

Mean PLR %	Slice Name	Static						Dynamic	
		SF7	SF8	SF9	SF10	SF11	SF12	Random	ADR
Static and Fixed Reservation	UCLE	17.99	17.90	17.99	19.74	25.37	31.24	18.73	14.29
	HCLE	26.98	26.91	25.97	24.21	25.98	27.36	27.75	18.75
	LCLE	55.03	55.20	56.04	56.05	57.27	59.18	53.52	32.17
Maximum Likelihood Estimation	UCLE	0.12	0.62	2.91	6.91	11.08	15.9	8.99	3.12
	HCLE	0.41	1.75	9.42	30.65	46.75	49.76	36.28	23.86
	LCLE	99.47	97.63	87.66	62.44	42.17	34.34	54.73	23.09
MBGD Prediction	UCLE	4.29	11.85	13.35	15.33	16.44	20.05	19.71	14.93
	HCLE	10.11	42.21	40.01	35.88	30.08	28.01	13.46	13.52
	LCLE	35.59	45.83	46.64	48.78	53.48	51.95	16.82	21.54

Table III.4: Packet Loss Rate Variation with various SF configurations

III.4.2 Applications Periodicity Impact

We study in this section the impact of load metric when fixe the number of LoRa devices assigned to each slice at 2500 devices. In this scenario, we focused on evaluating performance metrics in terms mean packet loss rate considering load in utilities computation. As showing in **Table III.5**, IoT device may have interests with the same gateway regardless of the slice that they occupy. However, these devices try to have the same bandwidth on this gateway, which lead to congestion case that affect decoding packet successfully. Therefore, in would be better decrease the load by distributing

and forwarding, less packet to another slice which less loaded on another gateway. In our simulation results summarized in the cited table, the 100 ms load configuration is reached the best performance. However, our simulation is continued by 100 ms.

Mean PLR %	Slice Name	Applications Periodicity					
		50	100	150	200	250	300
Static and Fixed Reservation	UCLE	28.36	18.95	35.58	16.66	16.66	16.66
	HCLE	11.07	29.23	6.53	4.18	3.33	0
	LCLE	27.22	51.80	24.54	0	0	0
Maximum Likelihood Estimation	UCLE	7.28	3.12	15.20	0	0	0
	HCLE	16.09	23.86	3.67	0	0	0
	LCLE	99.47	97.63	87.66	62.44	42.17	34.34
MBGD Prediction	UCLE	0	3.70	13.35	0	0	0
	HCLE	13.88	5.55	40.01	28.70	6	18.75
	LCLE	19.44	24.07	46.64	33.77	7.33	14.58

Table III.5: Packet Loss Rate Variation with multiple Applications Periodicity

III.4.3 Gateways Number and Positioning Impact

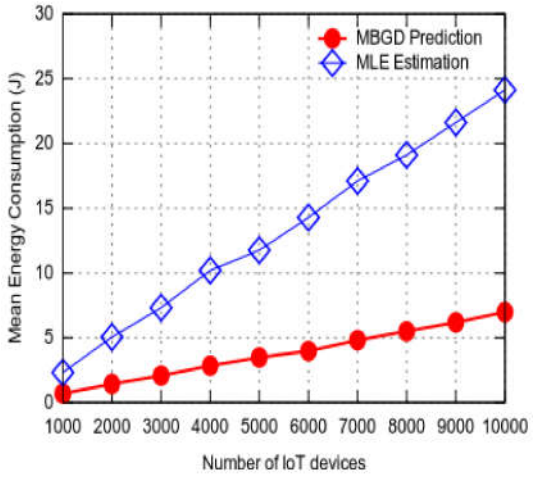
In **Table III.6**, we evaluate the impact of number of gateways deployed on each slice reliability with both estimation-based and prediction-based slicing algorithms. We evaluated the percentage of packet loss rate with 1, 4, and 16 GWs and we studied the result when we randomly placed the gateways. The result is next compared to a static positioning in which spaces between the gateways are all equal. With static positioning and when the number of gateways increase in the network, more channels are available for reservation for each virtual slice, this explains the observed improvement in reliability regardless of the adopted slicing strategies. Gateways positioning in an industry area is also very important, static positioning also had a better impact on reliability compared to the random positioning. However, there is still room for improvement because PLR was still high for LCLE slices and should be optimized with smart positioning algorithms.

Mean PLR %	Slice Name	Random Positioning			Static Positioning		
		1 GW	4 GWs	16 GWs	1 GW	4 GWs	16 GWs
Static and Fixed Reservation	UCLE	24.23	20.21	18.99	18.17	15.99	32.49
	HCLE	27.85	27.46	25.97	35.88	30.08	28.01
	LCLE	47.91	52.32	50.04	48.78	53.48	51.95
Maximum Likelihood Estimation	UCLE	0.12	0.62	2.91	6.91	11.08	15.9
	HCLE	0.41	1.75	9.42	30.65	46.75	49.76
	LCLE	99.47	97.63	87.66	62.44	42.17	34.34
MBGD Prediction	UCLE	7.45	11.85	13.35	15.33	0	0
	HCLE	42.40	42.21	40.01	15.99	13.90	0
	LCLE	50.15	45.83	46.64	32.49	16.44	20.05

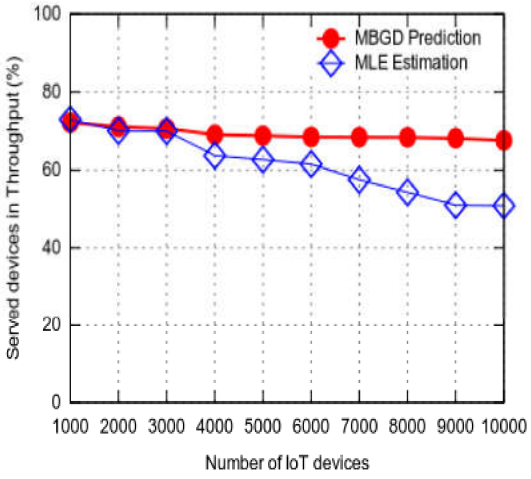
Table III.6: Packet Loss Rate Variation with multiple GW positioning configurations

III.4.4 Load impact on IIoT Network performance

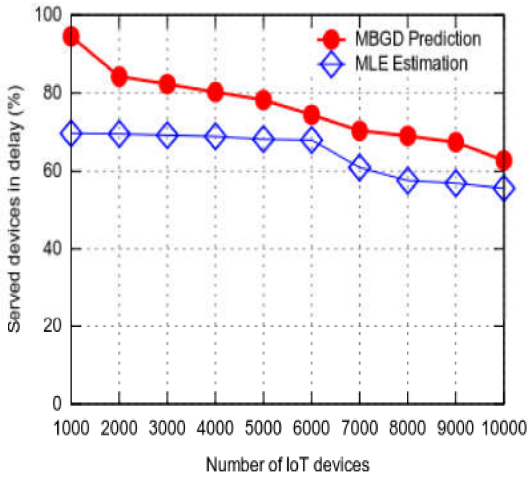
After evaluating the performance of static and dynamic slicing strategies in terms of configuration, applications periodicity and gateway positioning, we evaluate the performance of slicing strategies in congested IIoT deployments. We consider a large area of 10 KM radius and we increase the number of devices from 1000 to 10000 IoT objects (machine, robot). We evaluate how the performance varies in terms of QoS and energy consumption. The percentage of devices that respected delay and throughput thresholds are illustrated in **(Fig. III.5b)** and **(Fig. III.5c)** respectively. We also studied the impact on energy consumption in **(Fig. III.5a)** when the load in the area increases. On one hand, in an IIoT environment, devices were consuming more energy with estimation method when the number of devices increased in the network due to the slicing interval time between each optimization. It appears more beneficial to apply prediction to save more time when changing resource reservation for each virtual slice. A faster modification leaves a larger room to serve more devices and hence reduces the possibility of retransmiting packets and consumes less energy. On another hand, both MBGD prediction and MLE estimation slicing methods were quit efficient. However, MGBD had better performance than MLE and was less impacted in large scale IIoT deploy-



(a) Mean Energy Consumption



(b) Percentage of Served Devices in Throughput



(c) Percentage of Served Devices in Delay

Figure. III.5: Performance Evaluation with large scale IIoT Deployments

ments. This result is illustrated in **(Fig. III.5b)** which shows that when by passing 5000 devices in 10 KM square area, MLE estimation suffered from meeting throughput thresholds of LoRa virtual slices with a rate that nearly reached 50% for 10000 devices. This result is due to congestion when applied to a non-proper resource reservation. The latter is applied in a more efficient manner with MGBD prediction compared to the MLE estimation. In **(Fig. III.5c)**, the same analysis also goes for delay performance. Here, MBGD had the upper hand but both methods were impacted with congestion. MLE estimation was more impacted in large congestion and scored 60% for 10000 devices.

III.5 Conclusion

This chapter evaluates the dynamic prediction and estimation slicing strategies in large scale IIoT scenarios due to the rapid development of Industry 4.0. Meanwhile, it highlights the utility of supporting the adaptive dynamic slicing strategy with a slice-based parameters optimization that seeks for the best SF and TP configuration for each device depending on the slice that it belongs to. We compare a prediction-based slicing strategy to other estimation-based algorithm for inter slicing resource reservation. Both methods were deployed over a realistic IIoT scenario using NS3 simulator. MBGD prediction algorithm scored better results and was more efficient because it applies slicing decisions when considering QoS requirements of each slice. Results show the efficiency of MBGD in decreasing energy consumption and improving QoS, reliability when each device is configured with the proper SF and TP combination. Therefore, higher percentage of devices that respected delay and throughput thresholds. However, there is still room for improvement by optimizing the number of GWs and finding optimal positioning in an industry to provide more reliable communications for each LoRa virtual slice.

LoRaWAN will suffer from congestion when the number of devices increase in the network. Hence, we believe that centralized servers will practically face major difficulties in managing and properly isolating Lora slices as well as configuring each IoT devices with the proper parameters configuration. Hence, the goal in the next chapter is to propose a distributed strategy supported by SDN which should meet scalability and capacity requirements of LoRaWAN in large scale IoT deployments.

Chapter IV

Deep Federated Q-Learning-based Network Slicing for Industrial IoT

Summary

IV.1 Introduction

After evaluating in Chapter II the assets and the usability of network slicing in guaranteeing QoS for IIoT devices in terms of urgency and reliability, we have shown next, in Chapter III, that further improvement can be reached if an optimized SF and TP distribution is taken into consideration. However, due to the vast popularity that IoT is gaining, estimations forecast that 20 to 30 billion IoT devices will be connected by 2022 [40]. There is some doubts about how to deal with the rapid development of LoRaWAN knowing that the current LoRa architecture won't be capable of supporting upcoming scalability challenges in large scale LoRa deployments despite the advantages brought to LoRaWAN with our previous contributions.

Motivated by the IoT random access nature, network slicing is being investigated over LoRa technology in order to provide better isolation for multiple slice created on the unified physical LoRa devices. Therefore, these virtual slices are specified by their own QoS which will be independently and dynamically managed. However, due to the limited shared channel resources, slices implemented on LoRa devices suffer from performances degradation and resource starvation [36]. In this context, improving QoS for virtual LoRa slice requires an efficient and dynamic resource management scheme-based TP, SF, and bandwidth configuration. This remains as an open research issue.

Over the last years, network slicing has received increasing attention from the research community due to its several benefits in meeting QoS demands for slice's members. The authors in [89] proposed a framework based on Mini Batch GD and GMM to allocate channel resources for the slice member. This approach attempts to dynamically manage the QoS requirements, at the central SDN level, by checking the slice's throughput revenues. Authors in[36] proposed a LoRa network slicing scheme based on Maximum Likelihood Estimation to provide slice's resource reservation in inter and intra mode. This approach seeks to find the best Spreading Factor and Transmission Power configuration that provides an optimal decision on the channels number to be reserved. Authors in [56] proposed a hierarchical controller framework based on SDSense and defined the network component requirements, as well as deriving an algorithm to

optimise resource allocation across the network. In this approach, a logically centralised controller manages topology control, resource scheduling, while congestion control and data-rate reallocation modules react to the state of local controllers employed on each SDSense node. Authors in [128] proposed the adaptive control of the training iterations number under the resource-constrained edge environment. Moreover, in [120] the federated learning scheme over wireless networks context, was proposed, by formulating an optimization problem that find the optimal trade-off between computation and communication cost.

The ultra high density of IIoT network raises challenges in optimal wireless resource allocation which has been addressed by heuristic approaches because of its non-convex property [24]. Recently, deep learning approach-based resource allocation techniques are also proposed to provide efficient resource management, but the training data is either unavailable or the training process is computationally expensive and therefore are not suitable for large-scale system and cannot meet dynamic slice's QoS demands [48, 3, 60].

Reinforcement learning (RL) technique is able to adapt to the changes in dynamic environments. Therefore, it has been applied for resource scheduling [106], assignment optimization [141], etc. A RL agent is able to improve its policies by continuously interacting with the environment which can be formulated as a Markov Decision Process (MDP) [59]. Therefore, building high quality of policies in a centralized way is confronted with a great challenge when the state's features space are limited, the data privacy context, and the propagation delay [38]. To cope with these issues, Federated Learning (FL) has been proposed as a decentralized machine learning scheme. However, FL is designed to be a global-learning system. Therefore, local agent Q-network policy can utilize the agent's local computation capacity and share learning experience with other agents to train a global neural network model [72].

Unlike our previous contributions in Chapter II and Chapter III where we tackled resource reservation problem in centralized architecture. In this chapter, we will try to prevent potential challenges that may appear due to the increasing congestion in large scale IoT deployments by leveraging computational intelligence to LoRa gateways and moving closer to the edge. This should improve network performance and reduce

computational complexity in next generation IoT networks. Our main contributions with respect to the surveyed literature are stated as follows:

1. We propose a network slicing architecture based on SDN and NFV for IIoT 4.0 to meet multitude slice services requirements with guaranteed QoS for its members.
2. We formulate the LoRa slicing optimization model to be used in the proposed Framework.
3. We propose a Multi-Agent deep Q-learning (MAQL) based on network slicing paradigm to maximize self-QoS requirements.
4. We propose a Deep Federated Q-Learning (DFQL) scheme to federatively build global model that maximise slice's rewards (QoS), by improving action decision (TP and SF) and exploiting Agents self experiences.

The remainder of this chapter is organized as follow; section IV.2 provides the proposed slicing architecture, the system model, and the problem formulation. After that, we introduce, in section IV.3.1, our Deep federated Q-Learning framework in detail. Then we evaluate our scheme in section IV.4. Finally, section IV.5 concludes the chapter.

IV.2 Network Slicing Architecture-based System Model for IoT

In this section, we introduce the proposed architecture based on SDN and NFV to provide more flexibility in the management of IIoT networks and then we provide a deep overview of the slicing system model and the problem formulation.

IV.2.1 Network Slicing Architecture based SDN and NFV technologies

The design of the 5G network architecture is focused on careful consideration of the hardware infrastructure, software control and interconnectivity between them to ensure optimum slice management of the resources. In this context, we consider a network

slicing architecture that consists of a set of $J = \{1, ..., j\}$ IIoT network slices, where j represent the slice number. These slices are built on the unified physical infrastructure and share the same resources which consist of edge network and core networks.

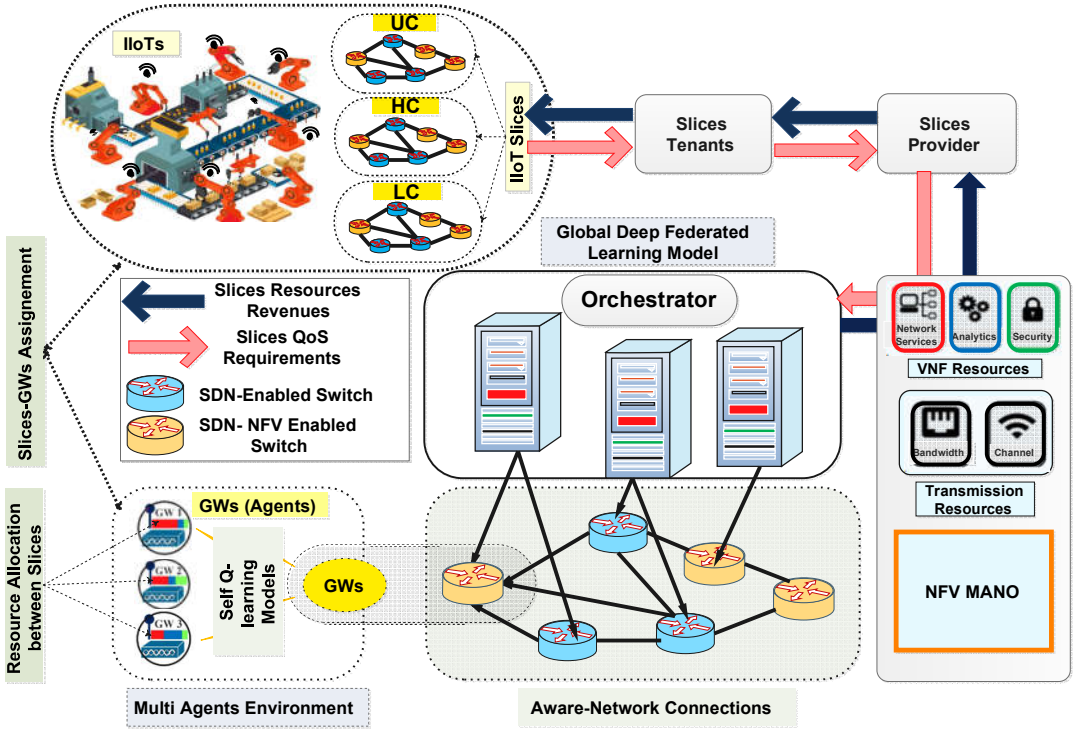


Figure. IV.1: Network Slicing Architecture.

Without loss of generality, the considered network slicing architecture, which is denoted in **Figure. IV.1**, consists of three virtual IIoT slices classified as follows; the most urgent slice denoted as the "Ultra high Critical of Latency and Efficiency (UCLE)" which yields more significance to the QoS, the reliability, and the efficiency. This explains the high demand for this slice by multitude industrial IoT safety applications such as emergency action and safeguarding systems. Thereafter, the "High Critical of Latency and Efficiency (HCLE)" slice donates less importance to the latency while maintaining reliability as a first target. This slice is required by the industrial scale reading applications. The last one is defined as the "Low Critical of Latency and Efficiency (LCLE)" which has the lowest slice priorities with unsecured QoS and performance. **Table IV.1** outlines the key QoS specifications of the slice in terms of latency, reliability, packet size, and priority factor [89].

These virtual IIoT slices are built on the top of the unified physical infrastructure layer that consist of a set of $K = \{1, ..., k\}$ gateways, where k is the number of gateways that will be considered as agents in our architecture. We emphasize that the involved resources go further than conventional equipment used. In addition to physical centralized and edge computing, it involves more manufacturing equipment with high sensing and actuation capacities, and network storage components. These agents play a key role, in the architecture, in providing resources to the substrate network layer, which contains a set of $I = \{1, ..., i\}$ IIoT devices, affected to the slices (virtual environment) that meet its QoS requirements. We consider the substrate network, in this contribution, as the real environment, in which agents play. All these agents interact with the SDN control unit, in which federated actions are provided based on their hierarchical learning experiences, to the local played model in order to equitably reserve physical resources between slices. While MANO is NFV's management and orchestration model which consist of a virtualized infrastructure manager, virtual network function management and orchestrator. It aims to provide a dynamic functions and network reconfigurability of the entire network.

Table IV.1: Key QoS Requirements of IIoT Network Slices

Slice Name	Packet Delay Budget (ms)	Reliability	Packet Size	Priority
UCLE	50	1-10^{-4}	24B	1
HCLE	100	1-10^{-4}	512B	2
LCLE	500	1-10^{-6}	250B	3

IV.2.2 Slicing System Model

This section provides the problem formulation and the system model of the slicing strategy in LoRa-based network. Two main components are considered in the proposed architecture. The distributed LoRa-gateways (agents) in the environment which provide self learning model and the SDN-based orchestrator which federatively manages the agent's resources based on their local model. The IIoT slices are integrated virtually on the top of these Gateways (GWs). However, GW's physical resources that include a set of $C = \{1, ..., c\}$ channels with each having a bandwidth $b \in B = \{1, ..., b\}$ on each GW $k \in K$, are virtually divided and reserved for each slice $j \in J$. To boost IIoT

communications, QoS requirements and conscious energy consumption must be considered for each slice member. Therefore, multi-objective optimization formulation that considers resource allocation in network slicing LoRa-based scenario is required.

The ultimate goal of this work is to provide dynamic channel allocation, based on TP and SF configuration, for IIoT slices member connected to a GW. We denote by $\alpha_i \in \{0, 1\}$ the binary variable to indicate the admission and the assignment success of device $i \in I$ to slice j on GW k. Therefore, we define the Throughput and Delay as in (IV.1) and (IV.2) respectively, based on SF and TP constraint for each device $i_{j,k}$ assigned to slice j on GW k [36].

$$\phi_i = SF.\frac{R_c}{2^{SF}}.CR = SF.\frac{b_{j,k}}{2^{SF}}.CR, \ \forall i \in I_{j,k}, \tag{IV.1}$$

$$d_i = \frac{L_i}{\phi_i}, \ \forall i \in I_{j,k}, \tag{IV.2}$$

where R_c, $b_{i,j}$, CR, and L_i denote respectively the chip rate, the bandwidth assigned for slice j on LoRa GW k, the coding rate, and the packet size. Where ϕ_i and d_i denote, respectively, the throughput, and the delay achieved by a device i assigned to slice j on GW k. Based on these formulas, we define in (IV.3) the QoS model that should be maximized for each slice on each LoRa GW.

$$max \ u_{QoS}^{j,k} = \sum_{i \in I_{j,k}} \alpha_i(\overline{\phi_i} + (1 - \overline{d_i})), \forall i \in I_{j,k}, \tag{IV.3}$$

where $u_{QoS}^{j,k}$ is denoted as a metric that yields indication about the slice's QoS satisfaction rate over a LoRa GW k, while $\overline{\phi_i}$ and $\overline{d_i}$ are denoted as a normalized value for throughput and delay respectively. Since the main objective is to meet slice's QoS requirements by improving resource demands, energy efficiency (EE) constitutes a second objective which must be maximized for slice members on each GW. However, EE aims to maximize the total communication rate within a unit energy. Let denote by p_i^t the power allocated for each device i assigned to slice j on GW k. Therefore, we define the EE system as a ratio of the slice's throughput sum to the total power consumption, which can be given

by (IV.4).

$$max\ u_{EE}^{j,k} = \sum_{i \in I_{j,k}} \alpha_i \frac{\overline{\phi_i}}{P_j^T + P_c}, \forall i \in I_{j,k}, \tag{IV.4}$$

where $u_{EE}^{j,k}$ denotes the EE metric that measures the slice's energy efficiency, P_c and $P_j^T = \sum_{i \in I_{j,k}} p_i^t$ denote the circuit power consumption and the TP respectively. However, best configuration on TP for devices leads to achieve better EE which improves the slice throughput. Increasing the SF also decreases the transmitted data rate, improves signal strength and gives the receiver a better sensitivity as stated in (IV.5).

$$p_i^r = \frac{p_i^t\ g_i^t\ g_i^r}{L} e^{\xi}, \forall i \in I_{j,k}, \tag{IV.5}$$

where p_i^r is the received power with g_i^r channel antenna gain, while g_i^t denote the channel antenna gain for the transmit power p_i^t. L is designated as the path loss depends on the distance between the transmitter and the receiver where e^{ξ} is the log-normal shadowing component with $\xi \sim N(0, \sigma^2)$. On the other hand, lower TP and SF configurations may cause packet reception failure consequent to the inter and intra-SF interference. The former presented as the Packet Loss Rate (PLR_i') resulting from collisions between two device configured with same SF. While the latter, denoted as binary value (PLR_i'') that stipulates, by following the hypothesis provided in [86], that the packet can survive interference from other LoRa GW transmissions. Thus, the devices undergo an estimated SINR value on the basis of (IV.6).

$$SINR_{i,j} = \frac{p_i^r}{\sigma^2 + \sum_{n \in \partial_j} p_n^r}, \tag{IV.6}$$

where p_i^r denotes the received power for packet n under consideration sent by device with $SF = i$ which ∂_j is the set of interfering packets with a common $SF = j$. Each element of the matrix downstairs designates the minimum signal power margin threshold $V_{i,j}$ with $i, j \in \{7, ..., 12\}$. It must be available for each packet to be sent, with $SF = i$, to be successfully decoded on each interfering packet with $SF = j$. Therefore, if a higher power margin (dB) value is achieved than the corresponding co-channel rejection value, the packet will outperform any interference packets, considering all SF combinations.

Therefore, we denote by (PLR_i''') the binary value that gives status whether the packet comes at the GW above or below sensitivity. In this context, we define Packet Success Rate (PSR_i), formulated in (IV.7), as the reliability (REL) objective that should be maximized for each slice on each GW.

$$V_{i,j} = \begin{pmatrix} & SF_7 & SF_8 & SF_9 & SF_{10} & SF_{11} & SF_{12} \\ SF_7 & -6 & 16 & 18 & 19 & 19 & 20 \\ SF_8 & 24 & -6 & 20 & 22 & 22 & 22 \\ SF_9 & 27 & 27 & -6 & 23 & 25 & 25 \\ SF_{10} & 30 & 30 & 30 & -6 & 26 & 28 \\ SF_{11} & 33 & 33 & 33 & 33 & -6 & 29 \\ SF_{12} & 36 & 36 & 36 & 36 & 36 & -6 \end{pmatrix}$$

$$\begin{aligned} max\ u_{REL}^{j,k} &= \sum_{j,k} \alpha_i PSR_i,\ \forall i \in I_{j,k}, \\ &= \sum_{j,k} \alpha_i((1 - PLR_i') + PLR_i'' + PLR'''), \end{aligned} \tag{IV.7}$$

where $u_{REL}^{j,k}$ denoted as the reliability metric that should be maximized for slices member on each LoRa GW.

At this stage and based on previously model mentioned, we define the multi-objective optimization problem, as in (IV.8), with the aim is to maximize the global slice utility revenues metric $U_{rm}^{j,k}$.

$$max\ U_{rm}^{j,k} = \sum_{j,k} (u_{QoS}^{j,k} + u_{EE}^{j,k} + u_{REL}^{j,k}), \forall k \in K, \forall j \in J, \tag{IV.8}$$

subject to the following constraint;

$$\sum \alpha_i = 1, \forall i \in I_{j,k}, \forall j \in J, \forall k \in K, \tag{IV.9a}$$

$$i_{j,k} \cap i_{j',k} = \oslash, \forall i \in I, \forall j, j' \in J, \forall k \in K, \tag{IV.9b}$$

$$i_{j,k} \cap i_{j,k'} = \oslash, \forall i \in I, \forall j \in J, \forall k, k' \in K, \tag{IV.9c}$$

$$0 < \sum \alpha_i p_i^T < p_j^T < p_k^{max}, \forall i \in I_{j,k}, \tag{IV.9d}$$

$$\sum \alpha_i \phi_i < \phi_{j,k}^{max}, \forall i \in I_{j,k}, \forall j \in J, \forall k \in K. \tag{IV.9e}$$

Since all defined IIoT slices are built on a common physical infrastructure and share the same resources, constraint IV.9a ensures that every i device is always admitted and allocated exactly to one and the only slice of the network that satisfies its QoS requirements, even if it has been built on different physical gateways. Additionally, constraints IV.9b and IV.9c ensure a perfect coalitions upon IIoT devices admission and association into virtual slices. However, IV.9b guarantees isolation between slices, in which the sets of two device assigned to the slice j and to the slice j', respectively, do not share common IIoT devices in the GW k range. While constraint IV.9c encloses a perfect isolation between LoRa GWs, in which devices that are assigned to slice j cannot be shared between two LoRa GWs $k, k' \in K$. Constraint IV.9d ensures that the transmission power $\sum \alpha_i p_i^T$ allocated to all members for each slice does not exceed the maximum transmission power p_j^T allocated for slice in turn does not exceed the maximum transmission power p_k^{max} provided by the GW k. Finally, IV.9e guarantees that the uplink traffic sum $\sum \alpha_i \phi_i$, sent through each gateway by slice members, do not exceed the maximum throughput capacity $\phi_{j,k}^{max}$ allocated for each slice.

Due to information deficiency between LoRa GWs in a large scale, a Deep Federated Q-learning (DFQL) is proposed to maximize the utility function revenues, based on TP and SF adjustment, on each GW and for each slice. **Table. IV.2** summarizes all key parameters used in this framework.

Table IV.2: Key Parameters Meaning

Notation	Meaning
$J = \{1, ..., j\}$	IIoT network slices set
$K = \{1, ..., k\}$	LoRa Gateways (Agents) set
$I = \{1, ..., i\}$	IIoT devices set associated to each slice
$C = \{1, ..., c\}$	LoRa-GW's channels set
$B = \{1, ..., b\}$	Channel Bandwidth set
$\alpha_i \in \{0, 1\}, \forall i \in I_{j,k}$	Device's admission and association index to slice
$\forall i \in I_{j,k}$	Device i assigned to slice j on Gateway k
SF	LoRa Spreading Factor
TP	LoRa Transmission Power
$\phi_i, \forall i \in I_{j,k}$	Throughput of device i
$d_i, \forall i \in I_{j,k}$	Delay of device i
$u_{QoS}^{j,k}$	Quality of Service metric for slice j on GW k
$p_i^t, \forall i \in I_{j,k}$	The power allocated for each device i
$u_{EE}^{j,k}$	Energy Efficiency metric for slice j on GW k
p_i^r	The received power
$V_{i,j}, \forall i, j \in \{7, ..., 12\}$	SF-Power Margin Matrix
$u_{REL}^{j,k}$	Reliability metric for slice j on GW k
$U_{rm}^{j,k}, \forall k \in K, \forall j \in J$	The global slice utility revenues metric
$\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}\}$	State, Action, Transition function , Reward
α and β	Agent α and Agent β
γ	Discount factor
$\theta_\alpha, \theta_\beta$	DQL Network parameters (weights)
θ_g	DNN Network parameters (weights)
$\mathcal{D}_\alpha, \mathcal{D}_\beta$	Reply memories to store transitions

IV.3 The proposed DFQL-based Network Slicing Framework

The proposed federated resource allocation-based SF and TP configurations are related to Multi-Agent deep Q-Learning (MAQL) which involves a set of agents (each one implements Deep Q-learning model) in a shared environment and a Global Deep Federated Learning (DFL) model plays an orchestrator role (SDN) for MAQL, as shown in **Figure. IV.2**. In this section, we provide, at the first stage, the problem formulation for the local resource allocation (for each GW) based on deep Q-learning (DQL) model. Then,

at the second stage, the deep federated Q-learning framework, for resource allocation, will be introduced.

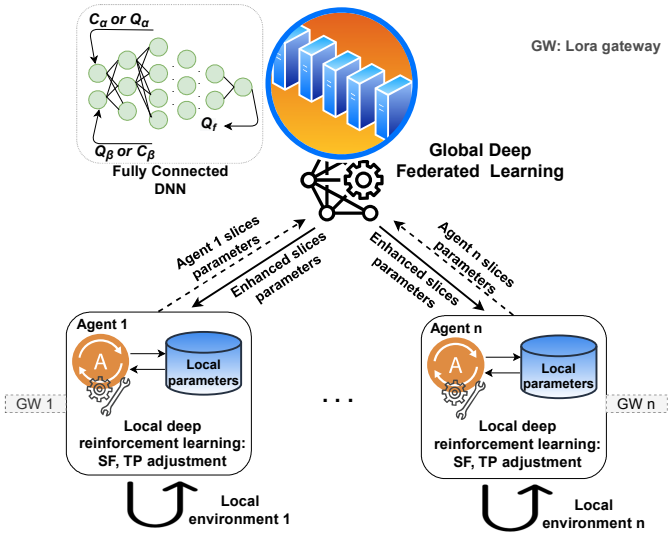


Figure. IV.2: Deep Federated RL Brain Architecture.

IV.3.1 Local DQL-based Slice's TP and SF Configurations

We propose to implement for each GW loRa a DQL-based slice's TP and SF configurations, as shown in **Figure. IV.3**. In this context, we define the local resource allocation based on DQL brain, that aims at maximizing slices profit, as a Markov Decision Process (MDP) [99]. However, MDP is defined on the basis of 4-tuple as $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}\}$. $\mathcal{S}$ is denoted as the state space indicating the virtual IoT slices state (EE, slice members), $\mathcal{A}$ is the set of action (TP and SF), $\mathcal{R}$ is the reward set indicating slices revenues in term of QoS. Where $\mathcal{T}$ represents the transition function, in which $\mathcal{T}(s, a, s') = P(s'|s, a)$ with $s' \in \mathcal{S}$ is the next state, $s \in \mathcal{S}$ given the current State, and $a \in \mathcal{A}$ denotes the action. We adopt, then, the relationship between $\{\mathcal{S}, \mathcal{A}, \mathcal{R}\}$ and the slicing metrics, in order to correlate network slicing optimization with DQL model, as follows:

$$s \in \mathcal{S} = \begin{pmatrix} EE_1 & I_1 \\ EE_2 & I_1 \\ EE_3 & I_2 \end{pmatrix}, a \in \mathcal{A} = \begin{pmatrix} TP_1 & SF_1 \\ TP_2 & SF_2 \\ TP_3 & SF_3 \end{pmatrix}, and$$

$$r \in \mathcal{R} = \begin{pmatrix} \phi_1 & d_1 \\ \phi_2 & d_2 \\ \phi_3 & d_3 \end{pmatrix}, \forall i \in I_{j,k}.$$

Where the numerations 1, 2 and 3 denote respectively UCLE, HCLE, and LCLE slices following its urgency factor.

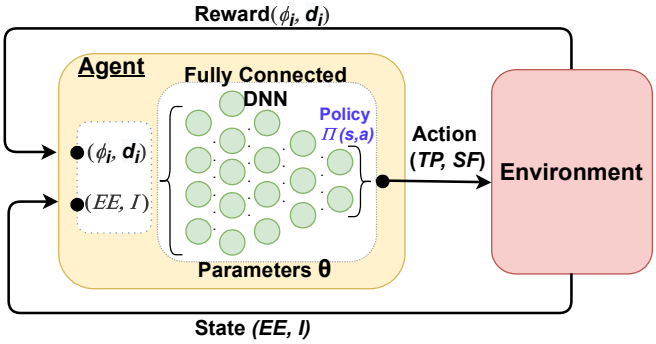


Figure. IV.3: Local DQL Model for resource allocation at LoRa GW level.

Therefore, two main equations in MDP process are defined as follow:

$$V_{t+1}^{\pi}(s) = r(s) + \sum_{s' \in S} T(s, \pi(s), s') V_t^{\pi}(s'), \tag{IV.10}$$

$$Q_{t+1}^{\pi}(s, a) = r(s) + \sum_{s' \in S} T(s, a, s') V_t^{\pi}(s'), \tag{IV.11}$$

the former $(V^{\pi}(s))$ denotes the function value, where $\pi : \mathcal{A} \rightarrow \mathcal{S}$ is the policy that represents the strategy used to select actions. The latter $(Q^{\pi}(s, a))$ is defined as Q-function value. However, solving the MDP problem means that choose the optimal policy value π^* by considering $V^{\pi^*}(s) = max_{\pi} V^{\pi}(s, \pi^*(s))$ or $Q^{\pi^*}(s) = max_{\pi} Q^{\pi}(s, \pi(s))$. While the Q-function value, for DQL model, in which the transition $\mathcal{T}$ is considered as unknown, is given by Q-network $Q(s, a; \theta)$, as in (IV.12).

$$Q_{t+1}(s, a; \theta) = \mathbb{E}_{s'}\{r(s) + \gamma \max_{a' \in \mathcal{A}} Q_t(s', a'; \theta) | s, a\}, \tag{IV.12}$$

where θ is denoted as the network parameter and γ is the discount factor. In this context, we define the replay memory $\mathcal{D}$ to store transitions $\{s, a, s', r\}$ which is used to learn

the network parameter by exploiting the Mini-Batch Gradient Descent process [89] that regularly update θ. Once the learning phase reaches the convergence, the optimal π^* policy is performed as in (IV.13).

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s, a; \theta) \tag{IV.13}$$

IV.3.2 The proposed DFQL framework

In this section, we present our DFQL framework in details. An overview of DFQL is shown in **Figure. IV.2**, in which the lower part constitutes the Multi-Agent deep Q-Learning (MAQL) and the upper part includes the Global Deep Federated Learning (DFL) model. Where the former (MAQL) assumes that agents do not share their partial observations but rather collaborate to receive global rewards, by sharing its local model based on Q-network experiences to the DFL orchestrator, rather than considering agent observations are shareable. While the latter (DFL) indicates that the orchestrator aggregates agent's self models and build a global deep network model to provide optimal action that maximize slices QoS revenues.

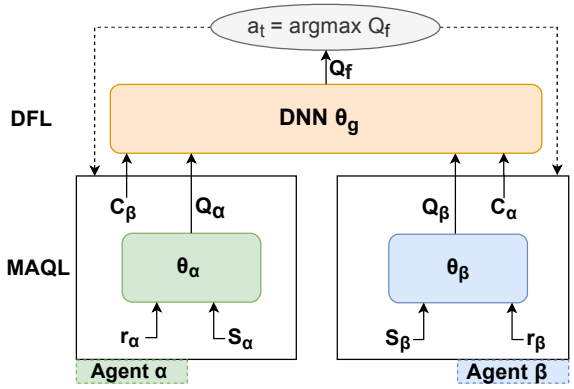


Figure. IV.4: Q-network Agents.

We assume that the MAQL consists of two agents, denoted by α and β, as shown in **Figure. IV.4**. However, we consider $\mathcal{D}_\alpha = \{s_\alpha, a_\alpha, s'_\alpha, r_\alpha\}$ and $\mathcal{D}_\beta = \{s_\beta, a_\beta, s'_\beta, r_\beta\}$ as two reply memories to store transitions parameters which will be collected to build federatively optimal policy (π^*_α and π^*_β) for Agents (LoRa GWs) α and β respectively. Respecting to agents α and β, the notations of Q-functions, states, actions, and policy

are denoted, respectively, as $\{Q_\alpha, s_\alpha \in \mathcal{S}, a_\alpha \in \mathcal{A}_\alpha, \pi^*_\alpha\}$ and $\{Q_\beta, s_\beta \in \mathcal{S}, a_\beta \in \mathcal{A}_\beta, \pi^*_\beta\}$. Thereby, assuming that states spaces (s_α and s_β), Transitions parameters ($\mathcal{D}_\alpha$ and $\mathcal{D}_\beta$), and the Q-network functions (Q_α and Q_β) are different for the defined agents α and β. Each agent (α or β) builds its own model based on self Q-network (Q_α or Q_β), and θ (θ_α or θ_β) parameters, as in **Figure. IV.4**.

These agents (MAQL part) interacts with the DFL global model (with Q-network Q_g and θ_g parameters) with the goal is to build a global federated model that satisfy dynamic slice's QoS requirements exploiting local agents experiences which are defined as $Q_\alpha(s_\alpha, a_\alpha; \theta_\alpha)$ from α and $Q_\beta(s_\beta, a_\beta; \theta_\beta)$ from β. In this context and based on MAQL Q-networks models, we define, in (IV.14), the DFL (based on DNN) Q-network output as $Q_f(\theta_\alpha, \theta_\beta; \theta_g)$.

$$Q_f(\theta_\alpha, \theta_\beta; \theta_g) = DNN([Q_\alpha(s_\alpha, a_\alpha; \theta_\alpha)|Q_\beta(s_\beta, a_\beta; \theta_\beta)]; \theta_g), \tag{IV.14}$$

where [.|.] denotes the concatenation notation and θ_g is the DNN (DFL) parameter which can be shared between agents.

Regarding the interaction with the DFL model, each agent contributes to build the global model, as well as federated Q-networks $Q_f(\theta_\alpha, \theta_\beta; \theta_g)$, by viewing the Q-network model of the other agent as a constant when updating its parameters and sending its local model to the global DFL orchestrator. Therefore, agent α sends its local model $Q_\alpha(s_\alpha, a_\alpha; \theta_\alpha)$ to the DFL to build the federated output Q^α_f, as in (IV.15), by considering agent's β model $Q_\beta(s_\beta, a_\beta; \theta_\beta)$ as a constant C_β. On the other hand, at its turn agent β sends its local model $Q_\beta(s_\beta, a_\beta; \theta_\beta)$ to build a global model Q^β_f, as in (IV.16), by considering agent's α local model $Q_\alpha(s_\alpha, a_\alpha; \theta_\alpha)$ as a fixed constant C_α.

$$Q^\alpha_f(s_\alpha, a_\alpha, C_\beta; \theta_\alpha; \theta_g) = DNN([Q_\alpha(.; \theta_\alpha)|C_\beta]; \theta_g), \tag{IV.15}$$

$$Q^\beta_f(s_\beta, a_\beta, C_\alpha; \theta_\beta; \theta_g) = DNN([Q_\beta(.; \theta_\beta)|C_\alpha]; \theta_g), \tag{IV.16}$$

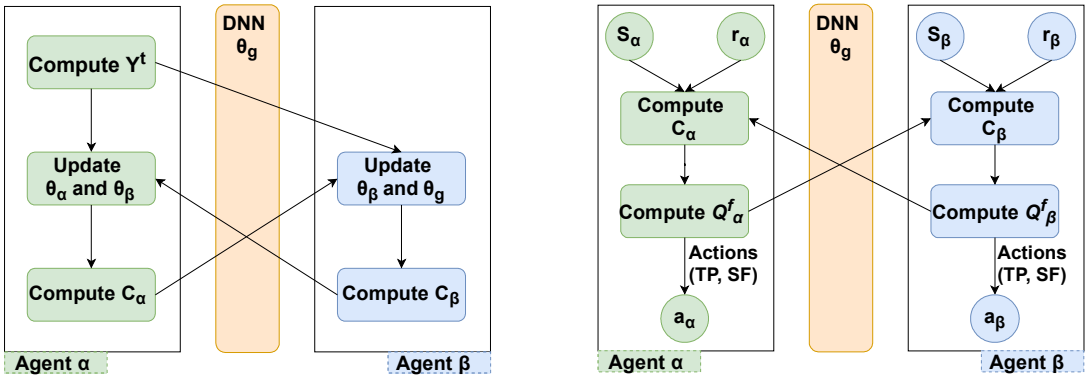
where $C_\alpha = Q_\alpha(s_\alpha, a_\alpha; \theta_\alpha)$ and $C_\beta = Q_\beta(s_\beta, a_\beta; \theta_\beta)$ noted as the defined constants used to update global Q-network model for agent β and α respectively.

At this stage, we define for agents α and β the Mean Square Error (MSE) Loss function denoted respectively in (IV.17) and (IV.18) [89]. These formulas are used to train our framework and update parameters $(\theta_\alpha, \theta_\beta, \theta_\beta)$ to build federated model and find an optimal action decision on TP and SF that maximize slice's QoS rewards then slice's metric (formulated in (IV.8)).

$$MSE^t_\alpha(\theta_\alpha, \theta_g) = \mathbb{E}[(Y^t - Q^\alpha_f(s^t_\alpha, a^t_\alpha, C_\beta; \theta_\alpha, \theta_g))^2] \tag{IV.17}$$

$$MSE^t_\beta(\theta_\beta, \theta_g) = \mathbb{E}[(Y^t - Q^\beta_f(s^t_\beta, a^t_\beta, C_\alpha; \theta_\beta, \theta_g))^2], \tag{IV.18}$$

where $Y^t = r^t(s) + \gamma \max_{a \in \mathcal{A}} Q^\alpha_f(s^t_\alpha, a^t_\alpha, C_\beta; \theta_\alpha, \theta_g)$ is considered for agent α only and then shared with agent β.



(a) Training phase (b) Testing phase

Figure. IV.5: Deep federated learning framework.

An overview of the training and testing phases of the DFQL framework is shown in **Figure. IV.5**. In the training phase, agent α, as a master, it begins by computing Y^t and sends it to agent β. This is accumulated as a first iteration to be used after in updating parameters. Agent β updates θ_β and θ_g and computes, then, C_β. After that, it sends θ_g and C_β to agent α. Subsequently, agent α updates θ_α and θ_g, computes C_α, and sends θ_g and C_α to agent β. **Figure. IV.5a** presents these training steps. In the testing phase, both agents compute C_α and C_β and send it to each other in order to compute federated output Q^α_f and Q^β_f, as provided in **Figure. IV.5b**.

We provide in Pseudo-code 7 and Pseudo-code 8 the detailed training phase for agent

Pseudo-code 7 Agent α Training Phase

Input : Slice State $\mathcal{S}_\alpha$, Action $\mathcal{A}_\alpha$, Reward $\mathcal{R}_\alpha$;
Output: θ_α, θ_g;

1 **begin**
2 Initialize: Q_α,Q_f based on random θ_α,θ_g, Memory D_α
for *Epoch=1:EP* **do**
3 **while** *MSE convergence=False* **do**
4 Collect $s_\alpha^t \in \mathcal{S}_\alpha$
Compute C_β=**ALG2.Compute**Q_β
Select $a_\alpha^t \in \mathcal{A}_\alpha$ with probability ϵ policy
Otherwise $a_\alpha^t = \arg\max_{a \in \mathcal{A}} Q_f^\alpha(s_\alpha^t, a_\alpha^t, C_\beta; \theta_\alpha, \theta_g)$
Apply a_α^t, get r_α^t and s_α^{t+1}
Store $s_\alpha^t, a_\alpha^t, r_\alpha^t, s_\alpha^{t+1}$ in D_β
Collect Batch-sample $(s_\alpha^j, a_\alpha^j, r_\alpha^j, s_\alpha^{j+1})$ of D_α
Compute C_β=**ALG2.Compute**$Q_\beta(j)$
$Y_\alpha^j = r_\alpha^j + \gamma \arg\max_{a \in \mathcal{A}} Q_f^\alpha(s_\alpha^j, a_\alpha^j, C_\beta; \theta_\alpha, \theta_g)$
Update θ_α,θ_g following formulas (IV.17) and (IV.18)
Compute $C_\alpha = Q_\alpha(s_\alpha^j, a_\alpha^j; \theta_\alpha, \theta_g)$
Computeθ_g=(**ALG2.Update**$Q(Y^j), C_\alpha; \theta_g)$
5 **end**
6 **end**
7 **end**

α and β and DFL (DNN). Firstly, these algorithms initialize memories ($\mathcal{D}_\alpha$,$\mathcal{D}_\beta$) and network parameters ($\theta_\alpha, \theta_\beta, \theta_g$) (line 2 in both algorithms). Then, in Pseudo-code 7 we collect state s_α^t and we call *Compute*Q_β function in Pseudo-code 8 to calculate the agent β Q-network output, obtain s_β^t, and select optimal a_β^t (line 2 to 9 in Pseudo-code 8). Therefore, from line 6 to 9 (Pseudo-code 7), the ϵ-greedy is performed for agent α, then transitions are stored in the memory. In steps 10 to 13 of Pseudo-code 7, the stored parameters are sampled and the *Compute*$Q_\beta(j)$ is called, from Pseudo-code 8 line 11, to calculate C_β output based on the index j. Thereafter, the lines 12 to 14 in Pseudo-code 7, agent α calculates Y_j, updates the parameters θ_α and θ_g, and computes C_α and send it to agent β. While in line 15 (Pseudo-code 7) the *Update*$Q(Y^j, C_\alpha; \theta_g)$ function of agent β is called to update θ_β ,θ_g, compute C_β as in line 16 to 19 in Pseudo-code 8. These steps will be repeated until convergence.

After the training phase, each agent (α, agents β) has its own model and network parameters, while the federated model (DFL) is now created. Over the test phase, the

Pseudo-code 8 Agent β Training Phase

Input : Slice State $\mathcal{S}_\beta$, Action $\mathcal{A}_\beta$, Reward $\mathcal{R}_\beta$;
Output: θ_β, θ_g;

begin
  Initialize: Q_β based on random θ_β, Memory D_β
  Function: ComputeQ_β
  for *Epoch=1:EP* **do**
    Collect $s_\beta^t \in \mathcal{S}_\beta$
    Select $a_\beta^t \in \mathcal{A}_\beta$ with probability ϵ policy
    Otherwise $a_\beta^t = \arg\max_{a_\beta \in \mathcal{A}} Q_\beta^t(s_\beta^t, a_\beta^t; \theta_\beta)$
    Apply a_β^t, get r_β^t and s_β^{t+1}
    Store $s_\beta^t, a_\beta^t, r_\beta^t, s_\beta^{t+1}$ in D_β
    return C_β=$Q_\beta(s_\beta^t,a_\beta^t;\theta_\beta)$
  end
  End Function

  Function: Compute$Q_\beta(j)$
  for *Epoch=1:j* **do**
    Collect Batch-sample $(s_\beta^j,a_\beta^j,r_\beta^j, s_\beta^{j+1})$ from D_β
    return C_β=$Q_\beta(s_\beta^t,a_\beta^t;\theta_\beta)$
  end
  End Function

  Function: Update$Q(Y^j, C_\alpha; \theta_g)$
  for *Epoch=1:j* **do**
    Collect Batch-sample (s_β^j,a_β^j) from D_β
    Update θ_β,θ_g following formulas (IV.17) and (IV.18)
  end
  End Function
end

agents α and β are now able to calculate and exchange C_α and C_β with each other through the global DFL in order to obtain federative outputs Q_α^f and Q_β^f, as shown in **Figure. IV.5b**. Thus, it will be exploited to find an optimal action decision on TP and SF configuration aimed at maximizing the slice's QoS rewards. As provided in Pseudo-code 9, federative communication done between agents via DFL model. From line 6 to 15 in this algorithm, agent α computes its own Q-network output (C_α) then sends it to the other agent through DFL. After that, based on C_β aggregated from agent β via DFL, it calculates its federative Q_α^f that will be used to find the optimal action

(on TP and SF) following slice's current states and rewards. While agent β does the same steps as α based on C_α provided in line 16 in Pseudo-code 9.

Pseudo-code 9 DFQL Scheme for Resources Allocation

Input : $s_\alpha^t \in \mathcal{S}_\alpha$, $s_\beta^t \in \mathcal{S}_\beta^t$, $a_\alpha^t \in \mathcal{A}_\alpha$, $a_\beta^t \in \mathcal{A}_\beta$, $r_\alpha^t \in \mathcal{R}_\alpha$, $r_\beta^t \in \mathcal{R}_\beta$;
Output: Agents Q_α^f, Q_β^f, Slice a_α^{t+1}, a_β^{t+1}, r_α^{t+1}, r_β^{t+1};
1 **begin**
2 Load Models: θ_α, θ_β, θ_g
  Initialize: Q_α^f, Q_β^f, Memory D_α and D_β
  for *Epoch=1:EP* **do**
3 **while** *Interaction=True* **do**
4 **Function:Agent-α to Agent-β via DFL (C_β)**
  Store C_β in D_α
  Collect $s_\alpha^t \in \mathcal{S}_\alpha$, $a_\alpha^t \in \mathcal{A}_\alpha$, $r_\alpha^t \in \mathcal{R}_\alpha$
  Compute C_α following formula (IV.12)
  Compute Q_α^f following formula (IV.15)
  Select: $a_\alpha^{t+1} \in \mathcal{A}_\alpha$ with probability ϵ policy
  Otherwise: $a_\alpha^{t+1} = \arg\max_{a \in \mathcal{A}} Q_f^\alpha(., C_\beta; \theta_\alpha, \theta_g)$
  Apply a^{t+1}, get r^t and s^{t+1}
  Store $a_\alpha^{t+1}, s_\alpha^{t+1}$, and r_α^t in memory D_α
  return C_α
  Function:Agent-β to Agent-α via DFL (C_α)
  Similar to *Function:Agent-α to Agent-β, (α must be replaced by β)*
  return C_β
5 **end**
6 **end**
7 **end**

IV.4 Experiment Results

This section provides the simulation results of the proposed DFQL framework for resource allocation. The parameters used in this simulation are summarized in **Table. IV.3**. Note that the proposed framework has been implemented in Python language using TensorFlow-gpu package on Intel Xeon E5-2620 v4 2x 8-Core with 64 GB RAM. Also, the NVIDIA GK110BGL [Tesla K40c] is used to improve speed during the training phase. In order to evaluate the proposed scheme, we examine firstly the training phase by analyzing MSE loss function and the success rate. Secondly, we analyze the energy consumption (EC) for slices. Then, we provide an evaluation of the slice's delay, the

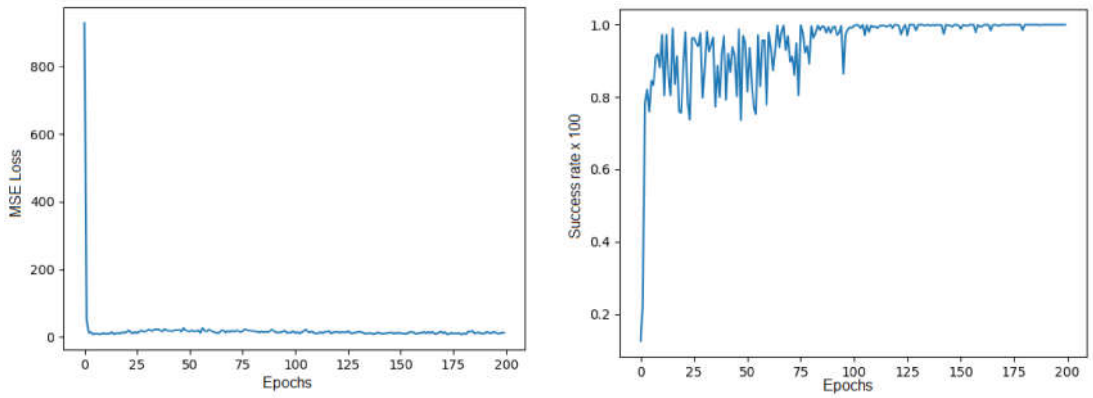
percentage of served devices, throughput, and the packet loss rate (PLR). Meanwhile, our framework will be compared with the centralized approach [89], given that it has the same IIoT context and the same QoS parameters used for the slices.

IV.4.1 Training and Loss Function (MSE) Evaluation

We study in this section the training phase performance of the proposed scheme. Loss function evaluation and the success rate are two ways to judge that the model is well trained. The former indicates, as shown in **Figure. IV.6a**, that the mini-batch gradient descent optimizer reach the convergence by tending to zero the loss function used to track the gradient error defined in IV.17 and IV.18. However, it indicates that model's weights (θ_α, θ_β, θ_g) are well tuned and ready to predict optimal outputs. While the success rate is defined as number of times (trials) the agent reached the end of path without colliding or reaching the time limit, to the total runs number because both timeouts and collisions are defined as failures. Where the trial is considered successful when s moves to a goal state within a defined time frame. The success rate result, shown in **Figure. IV.6b**, demonstrates that the learned policy is very efficient and powerful, attaining a success rate of almost 99.99%. This proves the robustness of the DFQL framework.

Table IV.3: Simulation parameters

Simulation Parameters	
LoRa GWs distribution	Constant position
Number of slice on each GWs	3
Number IIoT device Assigned to each slice	100 to 1000
Max Number of Channels per GWs	8
Channel Bandwidth	125KHz
Initial channel selection model	Uniformly distributed
European ISM sub-band	863 to 870 MHz
Spreading Factor set	{7,8,9,10,11,12}
Transmit power (TP) values set	2 to 14 dBm
Maximum Battery Capacity	950 mAh
DNN layers (DFL) {Input ;Hidden Layer; Output}	{2 ; 3,4,3,2 ; 1}
DNN layers (DQL) {Input ;Hidden Layer; Output}	{4 ; 5,4,3 ; 2}
Learning rate	0.001
Discount factor γ	0.05
Mini batch size	8
Epoch number	200
Iteration number per epoch	1000



(a) Loss Function Evaluation (b) Success Rate Evaluation

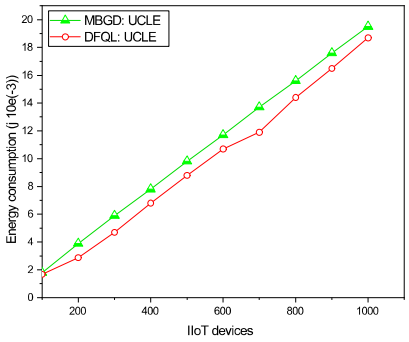
Figure. IV.6: Performances Evaluation

IV.4.2 Slices Energy Consumption Evaluation

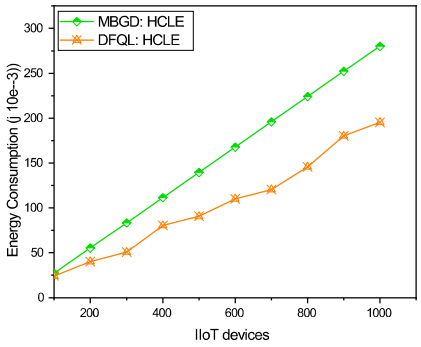
This section provides the EC evaluation for each slices on each GW LoRa. However, derived from equation (IV.4), we define the energy consumption model, implemented for this framework, as in formula (IV.19). Where T^{active} is denoted as the active time where the LoRa GWs interact in environment.

$$EC = \sum_{i \in I_{j,k}} \alpha_i (P_j^T + P_c) T^{active}, \forall i \in I_{j,k} \tag{IV.19}$$

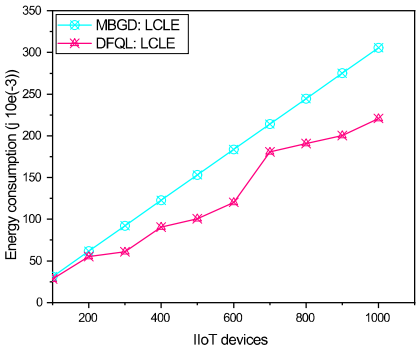
The EC in DFQL scheme is denoted as the mean EC for the same slice's type on agent α and β at many TP and SF configurations. As denoted in **Figure. IV.7**, EC metric is evaluated for DFQL and compared at the same time to the centralized approach based on machine learning tool (MBGD) provided in [89]. As noticed in **Figure. IV.7a, Figure IV.7b**, and **Figure. IV.7c**, the mean EC for UCLE, HCLE, and LCLE slice in DFQL approach increases when the number of deployed devices increase. However, always UCLE slice consumes less EC to the others even the IIoT devices reach a maximum number on GW k. It could be explained by the effect of SF and TP adjustments. When SF increases, the TP and the EC will increase subsequently for slices member. Therefore, following consideration of Energy Efficiency (EE) in-utility as a slice's State (s), agents will changes actions (a) on these parameters, applied on slices, to take the lowest SF and TP values which will drive device's delay-sensitivity to occupy the path



(a) UCLE EC: DFQL vs. MBGD



(b) HCLE EC: DFQL vs. MBGD



(c) LCLE EC: DFQL vs. MBGD

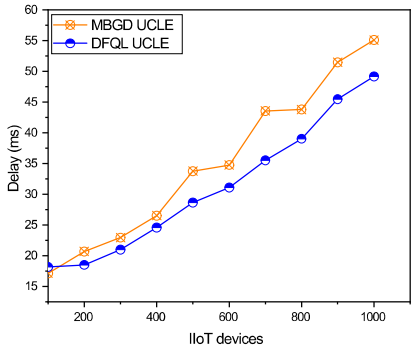
Figure. IV.7: Slices Energy Consumption Evaluation: DFQL VS. MBGD

with highest reliability compared to HCLE and LCLE slices. On the other hand, DFQL slices consume less EC than MBGD slices. This returns to the impact of the centralized approach compared to the federated scheme. Where, the former slice's requests are transferred to the centralized server and wait, to be served, a considerable time (LoRa GW active) that increases the slice's EC. While, for the latter approach, slices are served dynamically and federatively, on each Agent.

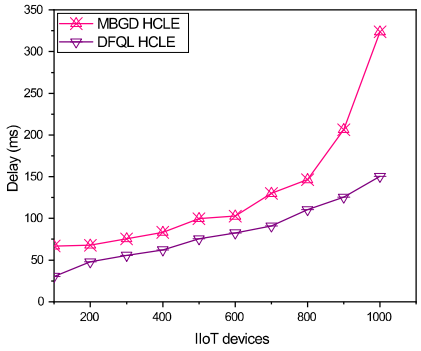
IV.4.3 Slice's Delay Evaluation

We analyse, in this section, the slice's delay of the proposed DFQL framework, as presented in **Figure. IV.8**, and then provides a comparison with the centralized approach provided in [89].

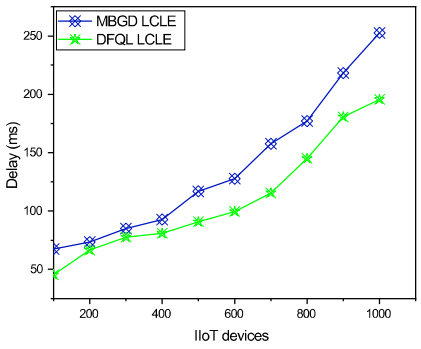
As denoted in **Table. IV.1**, slices are defined based QoS constraints. Among them,



(a) UCLE Delay: DFQL vs. MBGD



(b) HCLE Delay: DFQL vs. MBGD



(c) LCLE Delay: DFQL vs. MBGD

Figure. IV.8: Slices Delay variation Evaluation: DFQL VS. MBGD

delay plays an important role depending on the packet size parameter. Unlike LCLE, UCLE and HCLE slices are configured with the highest reliability and efficiency that expected to support the highest packet size transmission. It is remarkable, in **Figure. IV.8a**, **Figure. IV.8b**, and **Figure. IV.8c**, that the delay metric increases when increasing the number of devices assigned to each slice. However, always slice's UCLE delay slightly increases than the others, even if the IIoT devices reach a maximum number on GW k. This returns to the agent (α or β) that consider QoS slice's revenues as an important parameters to provide better action on SF and TP that maximizes, after some interaction, the rewards. Therefore, increasing the number of channel served to this slice that leads to maximize the throughput of slice's member that minimizes the mean delay. Noting that the slice's delay provided in the **Figure. IV.8** denotes the mean delay between the same slice's type of agent α and β. On the other hand, our

scheme outperforms the centralized approach that attempts to give much attention to the UCLE slice when the others wait a period of time to treat its requirements that increases congestion while the last devices were not served again. This impacts the mean served devices in delay, it increases the number of served devices in delay in our approach much than the MBGD scheme, as denoted in Figure IV.9.

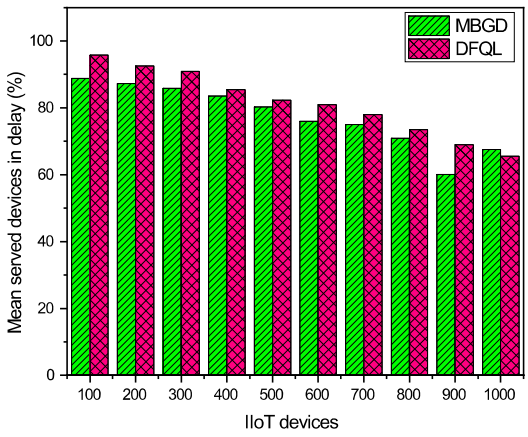
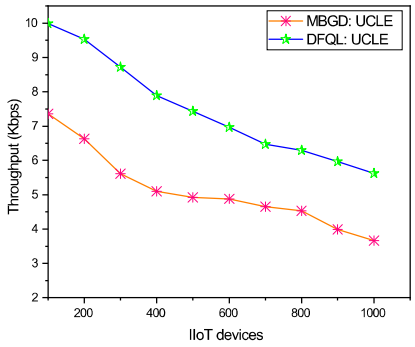


Figure. IV.9: Mean served devices in Delay: DFQL VS. MBGD.

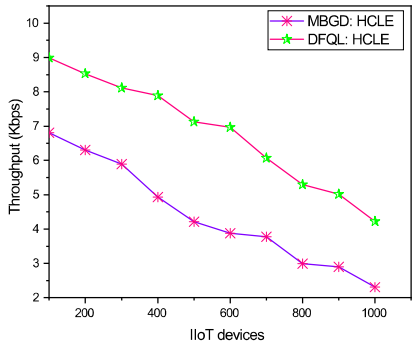
IV.4.4 Slices Throughput Evaluation

In this section we evaluate the mean throughput for slice's member, as denoted in **Figure. IV.10**, and compare them to the slice's throughput implemented within MBGD approach. Noting that the measure of throughput in DFQL denoted as the mean throughput for slice's members that have the same type (of agent (α, and β). It is remarkable that when increasing the number of deployed devices in each slice, the throughput decreases slightly. However, it returns to the augmented congestion in each slice. Regardless congestion effect, slices in DFQL framework have always the upper hand in throughput than the MBGD scheme, as provided in **Figure. IV.10a**, **Figure. IV.10b**, and **Figure. IV.10c**. While the UCLE slice outperforms all the coexist slices in throughput (DFQL). This due to the dynamic action provided by the federated framework that chooses a lowest SF action to configure slice's members when choosing the highest reliable paths between the ones available on each LoRa GW, which is not

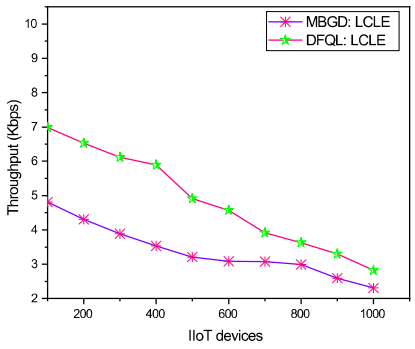
the case of the centralized approach that attempts to meet requirements in the central server.



(a) UCLE Throughput



(b) HCLE Throughput



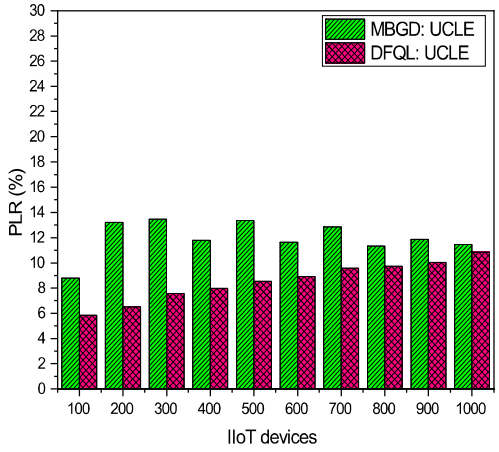
(c) LCLE Throughput

Figure. IV.10: Slices Throughput Evaluation: DFQL VS. MBGD

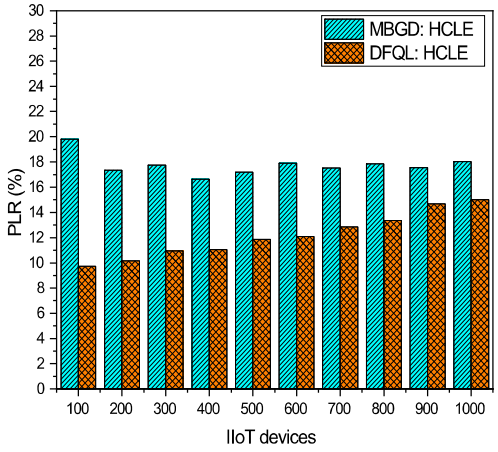
IV.4.5 Packet Loss Rate Evaluation

In this section we evaluate the mean percentage of PLR for slice's member, as denoted in **Figure. IV.11**, and compare them to the PLR of slices implemented within MBGD approach. Noting that the measure of PLR in DFQL denoted as the mean PLR for slices that have the same type (of agent (α, and β). As shown in **Figure. IV.11**, by increasing the IIoT devices, PLR will increase subsequently. This depends directly on throughput, that when increase it for devices much packet will be successfully transferred while it is not the case when throughput is low. We remark also, in **Figure. IV.11a**, **Figure. IV.11b**, and **Figure. IV.11c**, that UCLE and HCLE slices have

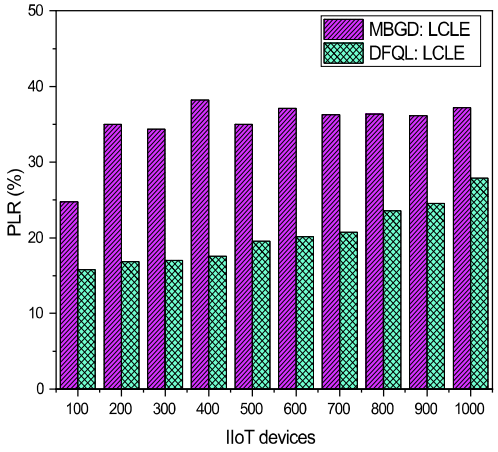
a reduced PLR compared to LCLE. However, this due to the reliability and efficiency consideration in this slice which is not the case in LCLE slice that consider only the load constraint. Compared the PLR slices result of DFQL to PLR slice results of MBGD, we could obviously note the efficiency of our proposed scheme in supporting dynamic slicing strategy by reducing PLR over than 9%. This improvement return to the shared federated learning and experience between agents that can improve the action decision making on TP and SF to slices.



(a) PLR of UCLE: DFQL vs. MBGD



(b) PLR of HCLE: DFQL vs. MBGD



(c) PLR of LCLE: DFQL vs. MBGD

Figure. IV.11: Slices PLR Evaluation: DFQL VS. MBGD

IV.5 Conclusion

LoRa is becoming a promising technology meeting IIoT network service requirements. With SDN, NFV, network slicing, and advanced deep reinforcement learning techniques, LoRaWAN enables flexible resource management and improves network performance in terms of energy consumption, reliability, and QoS. In this chapter, we proposed a novel federated network slicing based on deep reinforcement learning techniques for channels and bandwidth allocation. Each LoRa GW plays a role of an Agent in the environment and based on the learning experience provided by the other coexist agents via the global model, it virtually isolates and reserves its physical channels based on TP and SF adjustments in order to improve QoS rewards of each slice members. The numerical results obtained demonstrate the efficacy of our proposed framework, which outperforms other centralized slicing strategies because of its speed and its dynamism in serving slice's QoS requirements. Our future work is to explore the advantages of the proposed scheme in realistic testbed implementations.

General Conclusion and Perspectives

To support efficient IoT communications with guaranteed QoS requirements, new contributions are needed to provide flexible resource management in the network and optimize its configuration dynamically. In this thesis, we proposed new ideas that improve spectrum management, QoS consideration and energy efficiency in IoT networks using network slicing and software defined networking. The proposed solutions are implemented over LoRaWAN due its low power, wide area, open alliance and its potential to support large scale IoT deployments.

After dividing IoT services into three class of services based on urgency and reliability, network slicing is first implemented in the centralized SDN architecture where each class of services belongs to a network slice. The latter are proved to be completely isolated to protect urgent and critical IoT communications from being impacted by less prioritized IoT devices. Compared to the static configuration, results show that the dynamic-based prediction slicing was the best strategy in terms of QoS, reliability, and energy consumption. With this strategy, the centralized server defines, for each GW, how channels are reserved for the virtual slices configured on that GW based on devices throughput requirements estimation (using MBGD and MSE).

To improve these results, we extended the previous work (in chapter II) and made an in-depth performance evaluation and comparison with MLE-based prediction strategy. We adopted TOPG-based optimization that improves spreading factor and transmission

power parameters configuration at the physical layer for slice members. While respecting QoS thresholds of each slice, IoT devices are configured with *TOPG* to find the best TP and SF. Results show a major improvement in terms of QoS and energy consumption however, it is expected that the number of devices will increase over time and will exceed the maximum capacity that can be currently supported by a gateway which will, in turn, increase the rate of loss of packets in the different network slice. This challenge motivated us to look for an answer and find a solution to meet scalability challenges and guarantee QoS and energy efficiency LoRa devices.

Despite improving network performance with the previous contributions, LoRaWAN will still come up short in meeting scalability challenges in next generation large scale IoT networks. Therefore, we finally proposed an Deep SDN-based Federated LoRaWAN architecture (DFQL) based on multi-agent LoRa (MAQL) and slicing strategy improved reliability performance for IoT devices deployed in the network. When slicing and configuration decisions are leveraged to the edge (Agents), LoRa GWs will be able to apply the needed optimization faster instead of just sending all information to the server.

For future works, the focus should go towards investigating the findings of this thesis and implementing network slicing in real test-bed implementations. We will work on practically proving the slicing concept in all LoRa architecture layers. Moreover, some further improvements could be also realized by exploiting FPGA implementations to enable rapid analysis, prediction, and decision making. Finally, we also intend to work on other big challenges in IoT such as improving security measures and ensuring interoperability in next generation IoT networks.

Bibliography

[1] Ferran Adelantado, Xavier Vilajosana, Pere Tuset-Peiro, Borja Martinez, Joan Melia-Segui, and Thomas Watteyne. Understanding the limits of lorawan. *IEEE Communications Magazine*, 55(9):34–40, 2017.

[2] Erwin Adi, Adnan Anwar, Zubair Baig, and Sherali Zeadally. Machine learning and data analytics for the iot. *Neural Computing and Applications*, pages 1–29, 2020.

[3] Kazi Ishfaq Ahmed, Hina Tabassum, and Ekram Hossain. Deep learning for radio resource allocation in multi-cell networks. *IEEE Network*, 33(6):188–195, 2019.

[4] Mostafa Al-Emran, Sohail Iqbal Malik, and Mohammed N Al-Kabi. A survey of internet of things (iot) in education: Opportunities and challenges. In *Toward Social Internet of Things (SIoT): Enabling Technologies, Architectures and Applications*, pages 197–209. Springer, 2020.

[5] Bassel Al Homssi, Akram Ai-Hourani, Karina Gomez Chavez, Sathyanarayanan Chandrasekharan, and Sithamparanathan Kandeepan. Energy-efficient iot for 5g: A framework for adaptive power and rate control. In *2018 12th International Conference on Signal Processing and Communication Systems (ICSPCS)*, pages 1–6. IEEE, 2018.

[6] Akram Al-Hourani, Sithamparanathan Kandeepan, and Ekram Hossain. Relay-assisted device-to-device communication: A stochastic analysis of energy saving. *IEEE Transactions on Mobile Computing*, 15(12):3129–3141, 2016.

[7] Basim KJ Al-Shammari, Nadia Al-Aboody, and Hamed S Al-Raweshidy. Iot traffic management and integration in the qos supported network. *IEEE Internet of Things Journal*, 5(1):352–370, 2018.

[8] Fadi Al-Turjman, Hadi Zahmatkesh, and Ramiz Shahroze. An overview of security and privacy in smart cities' iot communications. *Transactions on Emerging Telecommunications Technologies*, page e3677, 2019.

[9] Gautami Alagarsamy, J Shanthini, and G Naveen Balaji. A survey on technologies and challenges of lpwa for narrowband iot. In *Trends in Cloud-based IoT*, pages 73–84. Springer, 2020.

[10] Furqan Alam, Rashid Mehmood, Iyad Katib, Nasser N Albogami, and Aiiad Albeshri. Data fusion and iot for smart ubiquitous environments: A survey. *IEEE Access*, 5:9533–9554, 2017.

[11] Mohammad Amiri-Zarandi, Rozita A Dara, and Evan Fraser. A survey of machine learning-based solutions to protect privacy in the internet of things. *Computers & Security*, page 101921, 2020.

[12] F Gregory Ashby and W Todd Maddox. Human category learning. *Annu. Rev. Psychol.*, 56:149–178, 2005.

[13] Paritosh Bahirat, Yangyang He, Abhilash Menon, and Bart Knijnenburg. A data-driven approach to developing iot privacy-setting interfaces. In *23rd International Conference on Intelligent User Interfaces*, pages 165–176, 2018.

[14] Theophilus Benson, Aditya Akella, and David A Maltz. Unraveling the complexity of network management. In *NSDI*, pages 335–348, 2009.

[15] Norbert Blenn and Fernando Kuipers. Lorawan in the wild: Measurements from the things network. *arXiv preprint arXiv:1706.03086*, 2017.

[16] Martin Bor and Utz Roedig. Lora transmission parameter selection. *International Conference on Distributed Computing in Sensor Systems*, 2017.

[17] Martin Bor, John Edward Vidler, and Utz Roedig. Lora for the internet of things. *International Conference on Embedded Wireless Systems and Networks*, 2016.

[18] Martin C Bor, Utz Roedig, Thiemo Voigt, and Juan M Alonso. Do lora low-power wide-area networks scale? In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 59–67. ACM, 2016.

[19] Chaillout Bouguera, Diouris and Andrieux. Energy consumption modeling for communicating sensors using lora technology. In *IEEE Conference on Antenna Measurements and Applications (CAMA)*. IEEE, 2018.

[20] Wolfgang Braun and Michael Menth. Software-defined networking using openflow: Protocols, applications and architectural design choices. *Future Internet*, 6(2):302–336, 2014.

[21] Leo Breiman et al. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, 16(3):199–231, 2001.

[22] Syed Hashim Raza Bukhari, Mubashir Husain Rehmani, and Sajid Siraj. A survey of channel bonding for wireless networks and guidelines of channel bonding for futuristic cognitive radio sensor networks. *IEEE Communications Surveys & Tutorials*, 18(2):924–948, 2015.

[23] Ben Buurman, Joarder Kamruzzaman, Gour Karmakar, and Syed Islam. Low-power wide-area networks: Design goals, architecture, suitability to use cases and research challenges. *IEEE Access*, 8:17179–17220, 2020.

[24] Stefano Buzzi, I Chih-Lin, Thierry E Klein, H Vincent Poor, Chenyang Yang, and Alessio Zappone. A survey of energy-efficient techniques for 5g networks and challenges ahead. *IEEE Journal on Selected Areas in Communications*, 34(4):697–709, 2016.

[25] Gustavo Carneiro. Ns-3: Network simulator 3. In *UTM Lab Meeting April*, volume 20, pages 4–5, 2010.

[26] Marco Cattani, Carlo Alberto Boano, and Kay Römer. An experimental evaluation of the reliability of lora long-range low-power wireless communication. *Journal of Sensor and Actuator Networks*, 6(2):7, 2017.

[27] Gilles Celeux and Gérard Govaert. Gaussian parsimonious clustering models. *Pattern recognition*, 28(5):781–793, 1995.

[28] Marwa Chafii, Faouzi Bader, and Jacques Palicot. Enhancing coverage in narrow band-iot using machine learning. In *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2018.

[29] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[30] Jiasi Chen and Xukan Ran. Deep learning with edge computing: A review. *Proceedings of the IEEE*, 107(8):1655–1674, 2019.

[31] Daniele Croce, Michele Gucciardo, Stefano Mangione, Giuseppe Santaromita, and Ilenia Tinnirello. Impact of lora imperfect orthogonality: Analysis of link-level performance. *IEEE Communications Letters*, 22(4):796–799, 2018.

[32] Daniele Croce, Michele Gucciardo, Ilenia Tinnirello, Domenico Garlisi, and Stefano Mangione. Impact of spreading factor imperfect orthogonality in lora communications. In *International Tyrrhenian Workshop on Digital Communication*, pages 165–179. Springer, 2017.

[33] Kajaree Das and Rabi Narayan Behera. A survey on machine learning: concept, algorithms and applications. *International Journal of Innovative Research in Computer and Communication Engineering*, 5(2):1301–1309, 2017.

[34] Anshuman Dash, Satyajit Pal, and Chinmay Hegde. Ransomware auto-detection in iot devices using machine learning. *International Conference on Business Analytics and Intelligence (ICBAI)*, 2018.

[35] Samir Dawaliby, Abbas Bradai, and Yannis Pousset. Adaptive dynamic network slicing in lora networks. *Future Generation Computer Systems*, 98:697–707, 2019.

[36] Samir Dawaliby, Abbas Bradai, Yannis Pousset, and Roberto Riggio. Dynamic network slicing for lorawan. In *Network and Service Management (CNSM), 2018 14th International Conference on*, pages 1–9. IEEE, 2018.

[37] Carmen Delgado, María Canales, Jorge Ortín, José Ramón Gállego, Alessandro Redondi, Sonda Bousnina, and Matteo Cesana. Joint application admission control and network slicing in virtual sensor networks. *IEEE Internet of Things Journal*, 5(1):28–43, 2018.

[38] John C Duchi, Michael I Jordan, and Martin J Wainwright. Privacy aware learning. *Journal of the ACM (JACM)*, 61(6):1–57, 2014.

[39] Mahmoud Elkhodr, Seyed Shahrestani, and Hon Cheung. The internet of things: new interoperability, management and security challenges. *arXiv preprint arXiv:1604.04824*, 2016.

[40] Ericsson. Internet of Things Forecast. https://www.ericsson.com/en/mobility-report/internet-of-things-forecast, 2019. [Online; accessed January-2019].

[41] TR ETSI. 103 467 v1.1.1:" speech and multimedia transmission quality (stq). *Quality of Service aspects for IoT*, 2018.

[42] Ivan Farris, Tarik Taleb, Yacine Khettab, and Jae Seung Song. A survey on emerging sdn and nfv security mechanisms for iot systems. *IEEE Communications Surveys & Tutorials*, 2018.

[43] John Fulcher. Computational intelligence: an introduction. In *Computational intelligence: a compendium*, pages 3–78. Springer, 2008.

[44] Yasser Gadallah, Mohamed H Ahmed, and Ehab Elalamy. Dynamic lte resource reservation for critical m2m deployments. *Pervasive and Mobile Computing*, 40:541–555, 2017.

[45] Salvador García, Sergio Ramírez-Gallego, Julián Luengo, José Manuel Benítez, and Francisco Herrera. Big data preprocessing: methods and prospects. *Big Data Analytics*, 1(1):9, 2016.

[46] Mouzhi Ge, Hind Bangui, and Barbora Buhnova. Big data for internet of things: A survey. *Future generation computer systems*, 87:601–614, 2018.

[47] Orestis Georgiou and Usman Raza. Low power wide area network analysis: Can lora scale? *IEEE Wireless Communications Letters*, 6(2):162–165, 2017.

[48] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[49] Claire Goursaud and Jean-Marie Gorce. Dedicated networks for iot: Phy/mac state of the art and challenges. *EAI endorsed transactions on Internet of Things*, 2015.

[50] Francis Griffiths and Melanie Ooi. The fourth industrial revolution-industry 4.0 and iot [trends in future i&m]. *IEEE Instrumentation & Measurement Magazine*, 21(6):29–43, 2018.

[51] IEEE 802 Working Group et al. Ieee standard for local and metropolitan area networks—part 15.4: Low-rate wireless personal area networks (lr-wpans). *IEEE Std*, 802:4–2011, 2011.

[52] GSMA. Network Slicing use cases requirements. https://www.gsma.com/futurenetworks/wp-content/uploads/2018/06/Network-Slicing-Use-Case-Requirements-_-FInal-.pdf, April 2018. [Online; accessed 19-august-2019].

[53] Renjie Gu, Shuo Yang, and Fan Wu. Distributed machine learning on mobile devices: A survey. *arXiv preprint arXiv:1909.08329*, 2019.

[54] Evangelos Haleplidis, Kostas Pentikousis, Spyros Denazis, J Hadi Salim, David Meyer, and Odysseas Koufopavlou. Software-defined networking (sdn): Layers and architecture terminology. Technical report, 2015.

[55] Yixue Hao, Daxin Tian, Giancarlo Fortino, Jing Zhang, and Iztok Humar. Network slicing technology in a 5g wearable network. *IEEE Communications Standards Magazine*, 2(1):66–71, 2018.

[56] Israat Haque, Mohammad Nurujjaman, Janelle Harms, and Nael Abu-Ghazaleh. Sdsense: An agile and flexible sdn-based framework for wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 68(2):1866–1876, 2018.

[57] Asma Haroon, Munam Ali Shah, Yousra Asim, Wajeeha Naeem, Muhammad Kamran, and Qaisar Javaid. Constraints in the iot: the world in 2020 and beyond. *Constraints*, 7(11):252–271, 2016.

[58] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Unsupervised learning. the elements of statistical learning (pp. 485-585), 2009.

[59] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, 2019.

[60] M Shamim Hossain and Ghulam Muhammad. A deep-tree-model-based radio resource distribution for 5g networks. *IEEE Wireless Communications*, 27(1):62–67, 2020.

[61] Jithin Jagannath, Nicholas Polosky, Anu Jagannath, Francesco Restuccia, and Tommaso Melodia. Machine learning for wireless communications in the internet of things: A comprehensive survey. *Ad Hoc Networks*, 93:101913, 2019.

[62] Madhab C Jena, Sarat K Mishra, and Himanshu S Moharana. Application of industry 4.0 to enhance sustainable manufacturing. *Environmental Progress & Sustainable Energy*, 39(1):13360, 2020.

[63] Devki Nandan Jha, Khaled Alwasel, Areeb Alshoshan, Xianghua Huang, Ranesh Kumar Naha, Sudheer Kumar Battula, Saurabh Garg, Deepak Puthal, Philip James, Albert Y Zomaya, et al. Iotsim-edge: A simulation framework for

modeling the behaviour of iot and edge computing environments. *arXiv preprint arXiv:1910.03026*, 2019.

[64] Tigang Jiang, Hua Fang, and Honggang Wang. Blockchain-based internet of vehicles: Distributed network architecture and performance analysis. *IEEE Internet of Things Journal*, 6(3):4640–4649, 2018.

[65] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[66] Anders Ellersgaard Kalør, René Guillaume, Jimmy Jessen Nielsen, Andreas Mueller, and Petar Popovski. Network slicing for ultra-reliable low latency communication in industry 4.0 scenarios. *arXiv preprint arXiv:1708.09132*, 2017.

[67] Anders Ellersgaard Kalør, Rene Guillaume, Jimmy Jessen Nielsen, Andreas Mueller, and Petar Popovski. Network slicing in industry 4.0 applications: Abstraction methods and end-to-end analysis. *IEEE Transactions on Industrial Informatics*, 14(12):5419–5427, 2018.

[68] Nei Kato, Bomin Mao, Fengxiao Tang, Yuichi Kawamoto, and Jiajia Liu. Ten challenges in advancing machine learning technologies toward 6g. *IEEE Wireless Communications*, 2020.

[69] Latif U Khan, Walid Saad, Zhu Han, and Choong Seon Hong. Dispersed federated learning: Vision, taxonomy, and future directions. *arXiv preprint arXiv:2008.05189*, 2020.

[70] Dae-Young Kim, Seokhoon Kim, Houcine Hassan, and Jong Hyuk Park. Adaptive data rate control in low power wide area networks for long range iot services. *Journal of computational science*, 22:171–178, 2017.

[71] Arun kishore Ramakrishnan, Davy Preuveneers, and Yolande Berbers. Enabling self-learning in dynamic and open iot environments. *Procedia Computer Science*, 32:207–214, 2014.

[72] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[73] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24, 2007.

[74] Rachel Kufakunesu, Gerhard P Hancke, and Adnan M Abu-Mahfouz. A survey on adaptive data rate optimization in lorawan: Recent solutions and major challenges. *Sensors*, 20(18):5044, 2020.

[75] Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of personalization techniques for federated learning. *arXiv preprint arXiv:2003.08673*, 2020.

[76] Xuan-Chien Le, Baptiste Vrigneau, Matthieu Gautier, Malo Mabon, and Olivier Berder. Energy/reliability trade-off of lora communications over fading channels. In *International Conference on Telecommunication*, 2018.

[77] Dong Li, Ming-Tuo Zhou, Peng Zeng, Ming Yang, Yan Zhang, and Haibin Yu. Green and reliable software-defined industrial networks. *IEEE Communications Magazine*, 54(10):30–37, 2016.

[78] He Li, Kaoru Ota, and Mianxiong Dong. Learning iot in edge: Deep learning for the internet of things with edge computing. *IEEE network*, 32(1):96–101, 2018.

[79] Lingling Li, Jiuchun Ren, and Qian Zhu. On the application of lora lpwan technology in sailing monitoring system. In *Wireless On-demand Network Systems and Services (WONS), 2017 13th Annual Conference on*, pages 77–80. IEEE, 2017.

[80] Qian Li, Geng Wu, Apostolos Papathanassiou, and Udayan Mukherjee. An end-to-end network slicing framework for 5g wireless communication systems. *arXiv preprint arXiv:1608.00572*, 2016.

[81] Shengyang Li, Usman Raza, and Aftab Khan. How agile is the adaptive data rate mechanism of lorawan? *arXiv preprint arXiv:1808.09286*, 2018.

[82] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[83] Jin-Taek Lim and Youngnam Han. Spreading factor allocation for massive connectivity in lora systems. *IEEE Communications Letters*, 22(4):800–803, 2018.

[84] M Carmen Lucas-Estañ, Miguel Sepulcre, Theofanis P Raptis, Andrea Passarella, and Marco Conti. Emerging trends in hybrid wireless communication and data management for the industry 4.0. *Electronics*, 7(12):400, 2018.

[85] Xiong Luo, Ji Liu, Dandan Zhang, and Xiaohui Chang. A large-scale web qos prediction scheme for the industrial internet of things based on a kernel machine learning algorithm. *Computer Networks*, 101:81–89, 2016.

[86] Davide Magrin, Marco Centenaro, and Lorenzo Vangelista. Performance evaluation of lora networks in a smart city scenario. In *Communications (ICC), 2017 IEEE International Conference on*, pages 1–7. IEEE, 2017.

[87] Mohammad Saeid Mahdavinejad, Mohammadreza Rezvan, Mohammadamin Barekatain, Peyman Adibi, Payam Barnaghi, and Amit P Sheth. Machine learning for internet of things data analysis: A survey. *Digital Communications and Networks*, 4(3):161–175, 2018.

[88] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.

[89] Seifeddine Messaoud, Abbas Bradai, and Emmanuel Moulay. Online gmm clustering and mini-batch gradient descent based optimization for industrial iot 4.0. *IEEE Transactions on Industrial Informatics*, 16(2):1427–1435, 2019.

[90] Konstantin Mikhaylov, Juha Petäjäjärvi, and Janne Janhunen. On lorawan scalability: Empirical evaluation of susceptibility to inter-network interference. In

Networks and Communications (EuCNC), 2017 European Conference on, pages 1–6. IEEE, 2017.

[91] Milan Milenkovic. *Internet of Things: Concepts and System Design.* Springer, 2020.

[92] RS Mitchell, JG Michalski, and TM Carbonell. *An artificial intelligence approach.* Springer, 2013.

[93] Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, and Mohsen Guizani. Deep learning for iot big data and streaming analytics: A survey. *IEEE Communications Surveys & Tutorials*, 20(4):2923–2960, 2018.

[94] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[95] Hussein Mroue, A Nasser, Sofiane Hamrioui, Benoît Parrein, Eduardo Motta-Cruz, and Gilles Rouyer. Mac layer-based evaluation of iot technologies: Lora, sigfox and nb-iot. In *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*, pages 1–5. IEEE, 2018.

[96] Tushar S Muratkar, Ankit Bhurane, and Ashwin Kothari. Battery-less internet of things–a survey. *Computer Networks*, 180:107385, 2020.

[97] Klara Nahrstedt, Hongyang Li, Phuong Nguyen, Siting Chang, and Long Vu. Internet of mobile things: Mobility-driven challenges, designs and implementations. In *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 25–36. IEEE, 2016.

[98] Akihiro Nakao, Ping Du, Yoshiaki Kiriha, Fabrizio Granelli, Anteneh Atumo Gebremariam, Tarik Taleb, and Miloud Bagaa. End-to-end network slicing for 5g mobile networks. *Journal of Information Processing*, 25:153–163, 2017.

[99] Almuthanna Nassar and Yasin Yilmaz. Reinforcement learning for adaptive resource allocation in fog ran for iot with heterogeneous latency requirements. *IEEE Access*, 7:128014–128025, 2019.

[100] Moises Nunez Ochoa, Arturo Guizar, Mickael Maman, and Andrzej Duda. Evaluating lora energy efficiency for adaptive networks: From star to mesh topologies. In *Wireless and Mobile Computing, Networking and Communications (WiMob),*, pages 1–8. IEEE, 2017.

[101] Nathalie Omnes, Marc Bouillon, Gael Fromentoux, and Olivier Le Grand. A programmable and virtualized network & it infrastructure for the internet of things: How can nfv & sdn help for facing the upcoming challenges. In *Intelligence in Next Generation Networks (ICIN), 2015 18th International Conference on*, pages 64–69. IEEE, 2015.

[102] Roger J Osborne and Merlin C Wittrock. Learning science: A generative process. *Science education*, 67(4):489–508, 1983.

[103] Juha Petäjäjärvi, Konstantin Mikhaylov, Marko Pettissalo, Janne Janhunen, and Jari Iinatti. Performance of a low-power wide-area network based on lora technology: Doppler robustness, scalability, and coverage. *International Journal of Distributed Sensor Networks*, 13(3):1550147717699412, 2017.

[104] Manoj Kumar Putchala. Deep learning approach for intrusion detection system (ids) in the internet of things (iot) network using gated recurrent neural networks (gru). 2017.

[105] Yongrui Qin, Quan Z Sheng, Nickolas JG Falkner, Schahram Dustdar, Hua Wang, and Athanasios V Vasilakos. When things matter: A survey on data-centric internet of things. *Journal of Network and Computer Applications*, 64:137–153, 2016.

[106] Chao Qiu, Haipeng Yao, F Richard Yu, Fangmin Xu, and Chenglin Zhao. Deep q-learning aided networking, caching, and computing resources allocation in software-defined satellite-terrestrial networks. *IEEE Transactions on Vehicular Technology*, 68(6):5871–5883, 2019.

[107] Fernando MV Ramos, Diego Kreutz, and Paulo Verissimo. Software-defined networks: On the road to the softwarization of networking. *Cutter IT journal*, 2015.

[108] Priyanka Rawat, Kamal Deep Singh, and Jean Marie Bonnin. Cognitive radio for m2m and internet of things: A survey. *Computer Communications*, 94:1–29, 2016.

[109] Brecht Reynders, Wannes Meert, and Sofie Pollin. Range and coexistence analysis of long range unlicensed communication. In *Telecommunications (ICT), 2016 23rd International Conference on*, pages 1–6. IEEE, 2016.

[110] Brecht Reynders, Wannes Meert, and Sofie Pollin. Power and spreading factor control in low power wide area networks. In *Communications (ICC), 2017 IEEE International Conference on*, pages 1–6. IEEE, 2017.

[111] Pedro HA Rezende and Edmundo RM Madeira. An adaptive network slicing for lte radio access networks. In *Wireless Days (WD), 2018*, pages 68–73. IEEE, 2018.

[112] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[113] Stuart J Russell and Peter Norvig. Artificial intelligence: a modern approach. malaysia, 2016.

[114] David R Shanks and Mark F STJOHN. Characteristics of dissociable human learning-systems. *Behav Brain Sci*, 17(3):367–395, 1994.

[115] Ali Asghar Nazari Shirehjini and Azin Semsar. Human interaction with iot-based smart environments. *Multimedia Tools and Applications*, 76(11):13343–13365, 2017.

[116] Sabrina Sicari, Alessandra Rizzardi, Luigi Alfredo Grieco, and Alberto Coen-Porisini. Security, privacy and trust in internet of things: The road ahead. *Computer networks*, 76:146–164, 2015.

[117] SA Sigfox. Sigfox technology overview, 2018.

[118] Mariusz Slabicki, Gopika Premsankar, and Mario Di Francesco. Adaptive configuration of lora networks for dense iot deployments. In *16th IEEE/IFIP Network Operations and Management Symposium (NOMS 2018)*, pages 1–9, 2018.

[119] Fengxiao Tang, Zubair Md Fadlullah, Bomin Mao, and Nei Kato. An intelligent traffic load prediction based adaptive channel assignment algorithm in sdn-iot: A deep learning approach. *IEEE Internet of Things Journal*, 2018.

[120] Nguyen H Tran, Wei Bao, Albert Zomaya, Nguyen Minh NH, and Choong Seon Hong. Federated learning over wireless networks: Optimization model design and analysis. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 1387–1395. IEEE, 2019.

[121] Mostafa Uddin, Sarit Mukherjee, Hyunseok Chang, and TV Lakshman. Sdn-based multi-protocol edge switching for iot service automation. *IEEE Journal on Selected Areas in Communications*, 2018.

[122] Nadège Varsier and Jean Schwoerer. Capacity limits of lorawan technology for smart metering applications. In *Communications (ICC), 2017 IEEE International Conference On*, pages 1–6. IEEE, 2017.

[123] Nuttakit Vatcharatiansakul, Panwit Tuwanut, and Chotipat Pornavalai. Experimental performance evaluation of lorawan: A case study in bangkok. In *Computer Science and Software Engineering (JCSSE), 2017 14th International Joint Conference on*, pages 1–4. IEEE, 2017.

[124] Benny Vejlgaard, Mads Lauridsen, Huan Nguyen, István Z Kovács, Preben Mogensen, and Mads Sorensen. Coverage and capacity analysis of sigfox, lora, gprs, and nb-iot. In *2017 IEEE 85th vehicular technology conference (VTC Spring)*, pages 1–5. IEEE, 2017.

[125] R Vinayakumar, KP Soman, Prabaharan Poornachandran, and S Sachin Kumar. Detecting android malware using long short-term memory (lstm). *Journal of Intelligent & Fuzzy Systems*, 34(3):1277–1288, 2018.

[126] Daiwat A Vyas, Dvijesh Bhatt, and Dhaval Jha. Iot: trends, challenges and future scope. *IJCSC*, 7(1):186–197, 2015.

[127] Meng Wang, Bo Cheng, Xuan Liu, Yi Yue, Biyi Li, and Junliang Chen. Poster: A sdn/nfv-based iot network slicing creation system. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, pages 666–668. ACM, 2018.

[128] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Ting He, and Kevin Chan. When edge meets learning: Adaptive control for resource-constrained distributed machine learning. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 63–71. IEEE, 2018.

[129] Antoine Waret, Megumi Kaneko, Alexandre Guitton, and Nancy El Rachkidy. Lora throughput analysis with imperfect spreading factor orthogonality. *arXiv preprint arXiv:1803.06534*, 2018.

[130] Gary White, Vivek Nallur, and Siobhán Clarke. Quality of service approaches in iot: A systematic mapping. *Journal of Systems and Software*, 132:186–203, 2017.

[131] Merlin C Wittrock. Learning as a generative process. *Educational psychologist*, 11(2):87–95, 1974.

[132] Andrew J Wixted, Peter Kinnaird, Hadi Larijani, Alan Tait, Ali Ahmadinia, and Niall Strachan. Evaluation of lora and lorawan for wireless sensor networks. In *SENSORS, 2016 IEEE*, pages 1–3. IEEE, 2016.

[133] Hao Wu, Kun Yue, Ching-Hsien Hsu, Yiji Zhao, Binbin Zhang, and Guoying Zhang. Deviation-based neighborhood model for context-aware qos prediction of cloud and iot services. *Future Generation Computer Systems*, 76:550–560, 2017.

[134] Bin Xiao. Self-evolvable knowledge-enhanced iot data mobility for smart environment. In *Proceedings of the 1st International Conference on Internet of Things and Machine Learning*, pages 1–14, 2017.

[135] Bin Xiao and Rahim Rahmani. A deep relation learning method for iot interoperability enhancement within semantic formalization framework. In *Proceedings of*

the Second International Conference on Internet of things, Data and Cloud Computing, pages 1–8, 2017.

[136] Bin Xiao, Rahim Rahmani, Yuhong Li, and Theo Kanter. Edge-based interoperable service-driven information distribution for intelligent pervasive services. *Pervasive and Mobile Computing*, 40:359–381, 2017.

[137] Mahmut Taha Yazici, Shadi Basurra, and Mohamed Medhat Gaber. Edge machine learning: Enabling smart internet of things applications. *Big data and cognitive computing*, 2(3):26, 2018.

[138] Volkan Yazıcı, Ulas C Kozat, and M Oguz Sunay. A new control plane for 5g network architecture with a case study on unified handoff, mobility, and routing management. *IEEE Communications Magazine*, 52(11):76–85, 2014.

[139] Zainab Zaidi, Vasilis Friderikos, Zarrar Yousaf, Simon Fletcher, Mischa Dohler, and Hamid Aghvami. Will sdn be part of 5g? *IEEE Communications Surveys & Tutorials*, 2018.

[140] Haijun Zhang, Chunxiao Jiang, Norman C Beaulieu, Xiaoli Chu, Xiangming Wen, and Meixia Tao. Resource allocation in spectrum-sharing ofdma femtocells with heterogeneous services. *IEEE Transactions on Communications*, 62(7):2366–2377, 2014.

[141] Yuan Zhou, Fuhui Zhou, Yongpeng Wu, Rose Qingyang Hu, and Yuhao Wang. Subcarrier assignment schemes based on q-learning in wideband cognitive radio networks. *IEEE Transactions on Vehicular Technology*, 69(1):1168–1172, 2019.

[142] Jiang Zhu, Yonghui Song, Dingde Jiang, and Houbing Song. A new deep-q-learning-based transmission scheduling mechanism for the cognitive internet of things. *IEEE Internet of Things Journal*, 5(4):2375–2385, 2017.

List of Publications

Journal Articles

Seifeddine Messaoud, Abbas Bradai, Olfa Ben Ahmed, Pham Quang, M Atri, and M Shamim Hossain. Deep federated q-learning-based network slicing for industrial iot. *IEEE Transactions on Industrial Informatics*, 2020.

Seifeddine Messaoud, Abbas Bradai, Syed Hashim Raza Bukhari, Pham Tran Anh Qung, Olfa Ben Ahmed, and Mohamed Atri. A survey on machine learning in internet of things: Algorithms, strategies, and applications. *Internet of Things*, page 100314, 2020.

Seifeddine Messaoud, Abbas Bradai, and Emmanuel Moulay. Online gmm clustering and mini-batch gradient descent based optimization for industrial iot 4.0. *IEEE Transactions on Industrial Informatics*, 16(2):1427–1435, 2019.

International Conferences

Seifeddine Messaoud, Abbas Bradai, and Mohamed Atri. Distributed Q-learning Based-Decentralized Resource Allocation for Future Wireless Networks. In *IEEE International Multi-Conference on Systems, Signals and Devices (SSD'20);**Accepted***, 2020.

Seifeddine Messaoud, Samir Dawaliby, Abbas Bradai, and Mohamed Atri. In Depth Performance Evaluation of Network Slicing Strategies in Large Scale Industry 4.0. In *IEEE International Multi-Conference on Systems, Signals and Devices (SSD'21); **Submitted***, 2020.

Book Chapters

Hajer Khlaifi, Samir Dawaliby, and **Seifeddine Messaoud**. Dynamic health assessment in water environments using lpwan technologies. *Wireless Technologies for Biomedical Applications (CRC Press, Taylor & Francis Group, P−BioMed, book series); **Accepted Book Chapter***, 2020.

Seifeddine Messaoud, Olfa Ben Ahmed, Abbas Bradai, and Mohamed Atri. Machine learning modelling-powered iot systems for smart applications. *Lecture Notes on Data Engineering and Communications Technologies (Springer Series); **Accepted Book Chapter***, 2020.

118