THESE DE DOCTORAT DE
SORBONNE UNIVERSITE

Spécialité
Informatique et Réseaux

Présentée par
M. KIM-HUNG LE

Pour obtenir le grade de
DOCTEUR de SORBONNE UNIVERSITE

Sujet de la thèse

Mécanismes d'interopérabilité pour les applications industrielles de l'Internet des Objets et la Ville Intelligente

Soutenue le 01 Avril 2019

Devant le jury composé de :

Prof. Christian Bonnet, Professeur, Eurecom, Sophia – France	Directeur de thèse
Prof. Paolo Papotti, Professeur, Eurecom, Sophia – France	Directeur de thèse
Prof. Karine Zeitouni, Professeur, Versailles – France	Rapporteur
Dr. Walid Dabbous, Directeur de Recherche, INRIA, Sophia – France	Rapporteur
Prof. Marcelo Dias de Amorim, Professeur, CNRS, Paris – France	Examinateur
M. Francois Hamon, Chercheur, Greencityzen, Marseille – France	Examinateur

INTEROPERATION MECHANISM FOR INDUSTRIAL INTERNET OF THINGS APPLICATIONS AND SMART CITY.

Kim-Hung Le

A doctoral dissertation submitted to:

Sorbonne University

In Partial Fulfillment of the Requirements for the Degree of:

Doctor of Philosophy

Specialty : Computer Science and Networking

Academic Supervisors:

Prof. Christian Bonnet	-	Eurecom, Sophia - France
Prof. Paolo Papotti	-	Eurecom, Sophia - France

Industrial Supervisor:

M. Francois Hamon	-	Greencityzen, Marseille - France

Acknowledgements

First of all, I would like to extend my sincere thanks to my advisors Prof. Christian Bonnet and Prof. Paolo Papotti for their valuable supports and brilliant ideas. Throughout this thesis, they have always spent the time in their busy schedule to guide and encourage my research activities. I also very much appreciate their competences that made this thesis work a success.

I would also like to thank M. Francois Hamon and M. Alexandre Boundone, my managers in Greencityzen company, who helped me shape my career and showed me how to transform my mistakes into skills. I really appreciate everything they helped me in both professional and personal life.

A special warm thank to M. Datta Soumya who helped me so much with his stimulating technical discussions and constructive publication reviewing. I am grateful to the committee members of my jury, Prof. Marcelo Dias de Amorim, Prof. Karine Zeitouni, and Dr. Walid Dabbous for their valuable inputs and time spent reading this thesis.

I would like to express my appreciation to my colleagues and friends at Greencityzen and Eurecom, for all the unforgettable enjoyable moments and their helps. I also wish to extend my warmest thanks to all my friends in France and Vietnam for all the wonderful time we spend together.

Finally, last but not least, I want to express my special gratitude to my parents, my fiancee for their unconditional support, love and trust. They, together with another members in my big family, make my life full of kindness and happiness with their encouragement.

Abstract

With the rapid growth of Internet technologies as well as the explosion of connected objects, Internet of Things (IoT) is considered an Internet revolution that positively affects several life aspects. However, in a large-scale deployment like a smart city scenario, billions of IoT devices generate a huge volume of data that must be processed. The integration of IoT solutions and cloud computing, namely *cloud-based IoT*, is a crucial concept to meet these demands. In this context, the enormous storage and computation capabilities make the cloud-based IoT a valuable solution to deal with a large amount of IoT data. However, two major challenges of the cloud-based IoT are interoperability and reliability. They come from the fact that IoT is typically characterized by heterogeneous devices, with constraints in storage, processing and communication capabilities. Moreover, there are no uniform standards in most IoT components such as devices, platforms, services, and applications.

In this thesis, our main objective is to deal with the interoperability and reliability issues that arise from large-scale deployment [1]. The proposed solutions spread over architectures, models, and algorithms, ultimately covering most of the layers of the IoT architecture. At the communication layer, we introduce a method to interoperate heterogeneous IoT connections by using a connector concept. We then propose an error and change point detection algorithm powered by active learning to enhance IoT data reliability. To maximize usable knowledge from this cleaned data and make it more interoperable, we introduce a virtual sensor framework that simplifies creating and configuring virtual sensors with programmable operators. Furthermore, we provide a novel descriptive language, which semantically describes groups of Things. To ensure the device reliability, we propose an algorithm that minimizes energy consumption by real-time estimating the optimal data collection. The efficiency of our proposals has been practically demonstrated in a cloud-based IoT platform of a start-up company.

[1]A huge number of devices including various device types are deployed over a large geographical area.

Contents

List of Figures

List of Tables

Glossary

List of Abbreviations and Acronyms

3GPP	3rd Generation Partnership Project
5G	Fifth Generation
API	Application Programming Interface
AR	Auto Regression
ARX	Auto Regressive with Exogenous Input
CEB	Cloud Edge Beneath
CoAP	Constrained Application Protocol
CoRE	Constrained RESTful environment
CSS	Cascading Style Sheets
DDL	Device Description Language
DNS	Domain Name Server
EWMA	Exponentially Eighted Moving Average
EXI	Efficient XML Interchange
GSN	Global Sensor Netwrok
HTML	Hypertext Markup Language
IaaS	Infrastructures as a Service
IEEE	Infrastructures as a Service
IERC	International Energy Research Centre
IETF	Internet Engineering Task Force
IMR	Iterative Minimum Repairing
IoE	Internet of Everythings
IoT	Internet of Things
IoT-VN	Internet of Things Virtual Network
IP	Internet Protocol
ITU	Committed to Connecting the World
JSON	JavaScript Object Notation
LDF	Logical Data Flow
LLAP	Lightweight Local Automation Protocol
LoRa	Short for Long Range
LPWAN	Low Power Wide Area Network
mDNS	Multicast Domain Name Server
MIoT	Massive Internet of Things
MQTT	Message Queuing Telemetry Transport

NanoIP	Nano Internet Protocol
NB-IOT	Narrow Band Internet of Things
NFV Network Function Virtualization	
NIST	National Institute of Standards and Technology
OASA	Online Adaptive Sampling Algorithm
OGC	Open Geospatial Consortium
OSGi	Open Service Gateway Initiative
OSI	Open Systems Interconnection Model
PaaS	Platform as a Service
RFID	Radio-frequency Identification
SASL	Simple Authentication and Security Layer
SDN	Software Defined Networking
SGS	Semantic Gateway as a Service
SMA	Simple Moving Average
SOA	Service Oriented Architecture
SSW	Semantic Sensor Web
sVSF	Scalable Virtual Sensor Framework
SWoT	Semantic Web of Things
TSMP	Time Synchronized Mesh Protocol
UNB	Ultra Narrow Band
UPnP	Universal Plug and Play
URL	Uniform Resource Locator
VoIP	Voice over Internet Protocol
VS	Virtual Sensor
VSE	Virtual Sensor Editor
W3C	World Wide Web Consortium
WoT	Web of Things
WoT-AD	Web of Things Asset Description
WoT-TD	Web of Things Things Description
WSN	Wireless Sensor Network
XML	Extensible Markup Language
XMPP	Extensible Messaging and Presence Protocol

1 Introduction

1.1 Motivation and Problem Statement

Internet of Things (IoT), also known as *Internet of Everything (IoE)*, is a novel paradigm that rapidly gains vast attention in the Internet era. The essential idea of IoT is the inter-networking of variety of "*Things*" through unique addressing schemes, where "Things" represent precisely identifiable objects [1] - such sensors, actuators, connected tags, and mobile phones. The IoT aims to provide a smart environment by bringing the Things from the physical world into the digital world. In other words, IoT is not only connecting the Things by using the Internet but it is also enabling data exchange [2]. Ideally, everyone can update in real-time the information or status of any Thing via IoT applications and services. If necessary, adaptive decisions according to predefined schemes are sent to control the Things. For example, in a smart office scenario, the temperature collected from the sensors and actuators in the office is sent to a collection point (e.g., gateway, central server), where the office workers could easily access via applications and services. If the temperature is higher than a defined threshold, a command will be automatically sent to the corresponding actuator to turn on the air conditioner.

It is not surprising that IoT is considered as a current revolution of the Internet that positively affects several real-life aspects. From an individual user view, the IoT enhances the life quality by offering several intelligent services. A typical example of such services is an intelligent transportation system. Vehicles and roads equipped with sensors and actuators could provide detailed information about the surrounding context to help drivers better navigate and thus improve safety [3]. Similarly, from a business point of view, automation, manufacturing, logistics, and transportation are areas where IoT reveals ideal opportunities to make current solutions more efficient and profitable [4]. For instance, in the logistics domain, every step of the supply chain from raw materials to commercial products can be monitored in real-time by using IoT technologies (e.g., connected tags, smart gateways). Thereby, enterprises can rapidly adapt to the changes in large markets. According to recent studies [5, 6], traditional businesses require at least 120 days to shift the supply chain while advance companies that have adopted IoT technologies (e.g., Metro, Wal-Mart) only need a few days. US National Intelligence Council listed IoT as one of the six "Disruptive Civil technology" that has been impacting the US economy [7]. The number of Internet-connected devices has exceeded the number of human beings on the planet in 2011. By the end of 2020, 212 billion IoT smart objects will be deployed worldwide, and by 2025, everyday Things, such as food containers, furniture, and paper documents, will be connected to Internet [8, 9].

Despite rapid growth and the increasing opportunities in everyday life aspects, IoT is still facing many critical challenges. Most of them relate to integrating, collecting, processing, and sharing the data from IoT Things in a large-scale deployment that has enormous device volume and various device types deployed over a large geographical area [10]. These challenges are caused by the fact that IoT is generally characterized by the massive number of heterogeneous Things, with constraints in storage, processing and communication capabilities. As a result, the raw data collected from such Things is huge, heterogeneous, and contains a vast amount of abnormal and redundant information [2]. Previous solutions that tackled similar challenges are no suitable for IoT scenarios due to two main reasons: (1) they are computationally expensive to operate on small Things; (2) they require human involvement, but it is impossible to involve humans in integrating billions of Things. Therefore, IoT service providers have been seeking innovative solutions to solve these challenges effectively. At the same time, *cloud computing* has emerged as a disruptive technology that provides extremely large storage and processing power capabilities. Exploiting these advantages may partially solve most of the IoT issues. For example, as constrained IoT devices cannot perform complex data processing, collected data can be transmitted to more powerful nodes (such as gateways, routers), but these nodes can not handle a large number of concurrent connections and their hardware is not dedicated to data processing. Thus, the scalability of this method is very limited. In this case, we may send this data to a cloud where the processing power, offered by a network of dedicated servers, is used to carry out the complex tasks from several devices. In addition, the cloud resources (such as cloud storage and processors) can be elastically provisioned and released to respond rapidly to different requests. Hence, cloud computing could solve the scalability issue in this example. Furthermore, IoT can leverage the huge storage capability from cloud computing to store IoT data more securely and in a way that is easier to access [11]. Therefore, the integration between IoT and cloud computing is inevitable to create a novel IT paradigm, namely *CloudIoT* or *cloud-based IoT* [12, 13, 14].

Although integrating IoT and cloud computing provides many benefits, the complex cloud-based IoT scenarios create several challenges that have received attention from research communities such as security and privacy, interoperability, big data, reliability, performance, and fog computing [15]. Two of those challenges targeted in our work are *interoperability* and *reliability*.

Interoperability: The most critical issue in cloud-based IoT is the lack of a unique standard in several elements from devices, platforms, services to applications [16]. Existing IoT standards are contributed from various *Standards Organizations* (such as Internet Engineering Task Force, Committed to Connecting the World) and scientific communities (such as World Wide Web Consortium, oneM2M) without coordination. In practice, it is extremely hard to integrate all existing contributions into a consistent, coherent one [17]. In addition, cloud-based IoT platforms have been typically designed as isolated vertical solutions for specific purposes [18]. For example, in a smart city scenario, each segment such as public transport, energy management, or water management, owns its particular platform, in which all components are tightly specialized in the application context [19]. To integrate with these platforms, the IoT providers must analyze in detail the requirements about hardware, software, and subsystems. Furthermore, new IoT device types along with their data format, which does not conform to the existing platforms, are emerging every day. These limitations lead to a substantial challenge about interoperability while deploying IoT solutions at large-scale. There is a clear need for unified architectures, mechanisms, and standard Things descriptions to facilitate the connection, transformation, and presentation of data from heterogeneous Things into usable knowledge [20].

Reliability: Many IoT applications (such as fire forest detection or earthquake prediction) are critical and strongly require the underlying technologies to be reliable. These applications must deliver high-quality services even in the presence of failures in the collected data. In addition, energy consumption of IoT devices highly impacts the overall system reliability. We can distinguish reliability in IoT into two parts as below:

- **Data reliability:** As consequences of rapid growth, there is a vast amount of raw data being continuously collected from billions of IoT devices. The flow of such data may widen human awareness about the natural phenomena, living environments or entities through intelligent services. However, this data is typically unreliable due to intense effects from many adverse factors including deployment scale [21], sensor constrained resources [22], and intermittent loss of connection [23]. Data reliability is uncritical in some cases, such as weather prediction and sentiment analysis that relies on overall data characteristics. In contrast, industrial applications, such as plant monitoring or fault detection, require strict data integrity and reliability. Missing values or outliers can trigger the wrong alerts or initiate an unneeded remedial process. Even in everyday applications, the failure to accurately indicate occupied parking, or full trash, may disrupt user experience and lead to trust issues with the IoT system. Therefore, maintaining data reliability is crucial for a successful IoT service deployment.
- **Device reliability:** A cloud-based IoT paradigm requires frequent data transmission from connected devices [24]. Such operations quickly drain the battery capacity and directly impact the device lifetime. In addition, a battery is limited in energy capacity. Recharging or replacing such battery is extremely costly or even impossible, because the IoT devices may be deployed in restricted environments (e.g., under the sewer networks or in the deep forest). Therefore, energy conservation techniques are crucial in the IoT services to achieve high reliability, especially in *low-power wide-area network* (LPWAN) scenario which stringently requires cost-effective and low-energy consumption [25]. Currently, most energy conservation techniques assume that data acquisition and processing consume lower energy than communication tasks [26]. Unfortunately, this assumption is incorrect for IoT scenario in which "power-hungry" sensors require high power resources to perform sensing tasks [27]. This triggers the complex issues that need to be solved to maintain device reliability for a long term.

In this thesis, our objective is to deal with interoperability and reliability issues raised in large-scale deployment. In other words, this research aims to produce IoT solutions that ensure:

- Maximizing IoT interoperability in different levels from connectivity, data format to IoT application;
- Minimizing the abnormality in collected data;
- Maximizing the valuable knowledge from collected data;
- Minimizing the energy consumption in the IoT devices while maintaining high service quality.

We use the traditional Internet of Things Architecture (IoT-A) to illustrate our contributions in Figure 1.1. At the communication layer, we introduce a method to minimize the effort to establish and configure the connectivity to heterogeneous IoT Things by automatically generating connectors (number 1 in the figure). These connectors also speed up the

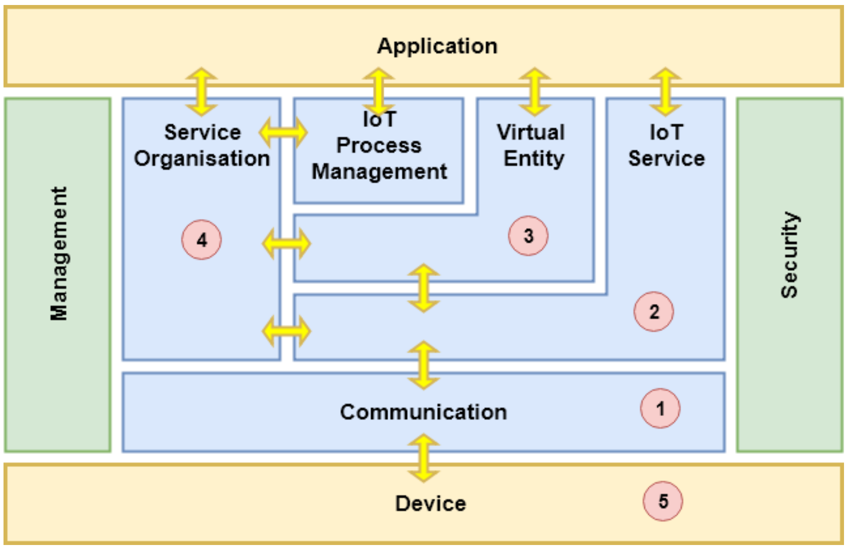


Figure 1.1 – Our contributions positioned in the IoT functional view

data acquisition from open data sharing web services. To minimize the error in collecting data from IoT devices, we propose an error and change point detection algorithm powered by active learning as an IoT service (number 2 in the figure). After cleaning data, we try to maximize usable knowledge from this data by proposing a virtual sensor framework that simplifies creating and configuring virtual sensors with programmable operators such as rules, formulas, and functions (number 3 in the figure). To increase the interoperability for IoT applications, we provide a descriptive language, which semantically describes not only single Things but also group of Things (number 4 in the figure). At the device side, we propose an algorithm minimized energy consumption by real-time estimating the optimal data collection frequency based on historical data (number 5 in the figure). This algorithm is experimentally proved to increase IoT device reliability significantly. In summary, our contributions spread over architecture, models, and algorithms and cover most of the layers of the IoT architecture.

1.2 Thesis Contributions and Outline

The key contributions in this thesis can be summarized as follows:

IoT Interoperability Solutions

- *A method to interoperate IoT device connections using connectors*: This solution represents an innovative IoT framework that could facilitate creating cloud-based connectors for heterogeneous connections from IoT Things. Typically, the connector is a specific code segment used to wrap connectivity objects in a simple RESTful web service. Thereby, the end-user could easily connect to heterogeneous devices via the web service regardless of the complexity behind. Furthermore, our proposal may assist end-users in quickly retrieving data from various data sources via the connectors created from given templates. The interoperability with other implementations is preserved by using ongoing IoT standardization methods.
- *An IoT Framework to maximize usable knowledge from IoT data using virtual*

sensors: This framework simplifies creating and configuring *virtual sensors* (VSs) with programmable operators, such as rules, formulas, or functions. These VSs could be linked together to create a topology, namely *logical data-flow* (LDF), that enables producing high-level information from collected data. The outcomes of LDF are formed under *JavaScript Object Notation for Linked Data* (JSON-LD) format, which is used to generate interpretable data across different IoT platforms [28]. In this way, it significantly increases the interoperability of our solution. On the top of the framework, a *Virtual Sensor Editor* (VSE) is implemented to facilitate building and configuring the LDF by offering the drag-drop actions on a web interface. To increase scalability and performance, we implement our proposal based on clustering architecture along with various strategies, such as executing LDF following asynchronous model and managing a non relational database to store and query data.

- *A semantically Descriptive Language for Group of Things*: Our proposal allows to semantically present compound objects, namely *Assets*, in Massive IoT scenario. To achieve this goal, we introduce a novel semantic description named *Web of Things – Asset Description* (WoT-AD) and a light-weight Web of Things framework, fully integrated together for maximizing the interoperability. Such combination is not only capable of presenting, accessing, and managing the Asset but also speeding up the IoT application development. The effectiveness of designed solution is ensured by choosing and combining ongoing technologies and IoT frameworks that have been practically demonstrated in real use-cases.

IoT Reliability Solutions

- *An Active Learning method for errors and events detection in time series.*: In this work, we exploit active learning to detect both errors and events in a single algorithm while minimizing user interaction. For the detection, we introduce a non-parametric algorithm, which accurately detects anomalies exploiting a novel concept of neighborhood and unsupervised probabilistic classification. Given a desired quality, the confidence of the classification is then used as termination condition for the active learning algorithm. Experiments on real and synthetic datasets demonstrate that our approach achieves high performance (F-score) by labeling a very limited number of data points. We also show the superiority of our solution compared to the state-of-the-art approaches.
- *An Energy-efficient Sampling Algorithm*: The proposed algorithm minimizes the energy consumption of IoT devices by estimating the optimal data collection frequency in real-time based on historical data. Practical experiments have shown that the proposed algorithm can reduce the number of acquired samples up to four times in comparison with a traditional fixed-rate approach. In addition, our proposal is light-weight enough to be deployed on constrained-resource IoT devices.

The work presented in the thesis is structured as follows. Apart from the introduction and conclusion, we divide the main content into three parts. We present the background knowledge and related works in the first part. Then, in part II and III, we discuss the solution for the interoperability and reliability issues in IoT, respectively.

1. In the first part, we provide the fundamental knowledge of IoT, especially the cloud-based IoT paradigm. This part also highlights the current issues and challenges relating to interoperability and reliability in IoT. Then, the existing approaches to address

these challenges are analyzed in detail to show their advantages and limitations. Base on this analysis, we introduce our solutions in the following parts.

2. The second part discusses several issues regarding Interoperability in IoT. Chapter 4 presents a solution to facilitate the connectivity to heterogeneous IoT Things. Chapter 5 introduces the solution to maximize usable knowledge using virtual sensors. Chapter 6 discusses the interoperability for IoT Things and proposes a descriptive language to semantically describe not only single Things but also group of Things.

   Results have been presented and/or published

   (a) At 3rd IEEE International Forum on Research and Technologies for Society and Industry 2017 (RTSI) [29].

   (b) At 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) [30].

3. In the last part, we focus on the solutions to increase the reliability in IoT. In chapter 7, we present an errors and change point detection algorithm using active learning. Then, in chapter 8, we propose an adaptive sampling algorithm to minimizing the energy consumption in IoT devices.

   Results have been submitted as patents:

   (a) Procede pour detecter distinctement des anomalies isolees, des anomalies collectives et des points de ruptures et dans une serie temporelle de mesures capteur, 2018, Patent No L. 612-2 R.612-8, France.

   (b) Procede pour reduire la consommation d‘energie ainsi que la frequence de mesure et de transmission d’un capteur connecte (submitted).

Part I

Background Analysis

2
Reference Technologies

2.1 Internet of Things Overview and Related Concepts

2.1.1 Internet of Things

The concept of Internet of Things was first coined by Kevin Ashton, Executive Director of the Auto-ID Center in Massachute Institute of Technology in 1999, and it presents a world where billions of objects can sense, communicate, and share information [31]. Subsequently, IoT becomes more popular due to the explosion of mobile devices, ubiquitous communication, and cloud computing. However, the definition of "Internet of Things" is still ambiguous and has various facets depending on different perspectives. From functional points of view, IoT is defined as "Things has identities and virtual personalities operating in smart space using intelligent interfaces to connect and communicate within the social, environmental, and user contexts" [32]. Semantically, the term of Internet of Things is composed of two words "Internet" and "Things". Following this way, IoT is defined as "a worldwide network of interconnected objects uniquely addressable, based on standard communication protocols" [33].

Similar to its definition, the characteristics of IoT vary from one domain to another. Some key characteristics are identified during our research study as following:

- **Dynamic and Self-adapting**: IoT devices and systems must be able to dynamically adapt to the changes in contexts or sensed environments. For example, considering a smart building system comprising of many temperature devices, these devices can adapt their modes based on deployed scenarios (e.g., indoor or outdoor). Thereby, they can adjust their calibrations to collect data more accurately.
- **Self-configuring**: IoT devices need to self-configure about connectivity, software upgrades, and sensor drivers. This ability allows managing a large number of devices to provide common services.
- **Interoperable Communication Protocols**: IoT devices have to support multiple connectivity technologies (e.g., Wifi, Lora, Sigfox) to communicate with other devices and cloud infrastructures.
- **Unique Identify**: IoT devices must be identified by an unique identity (such as an *Internet Protocol* (IP) address or an *Uniform Resource Locator* (URL)). In addition, the IoT system needs providing interfaces (Application Programming Interfaces (API), web interfaces), which allow the end-user to access directly to the device resources.

- **Integrated into Information Network**: To communicate and exchange data, IoT devices must be integrated into information networks. In addition, these devices have to be dynamically described and discovered by other IoT devices or systems in the same network. Such integration may enrich the acquired information because of the data aggregation from several nodes.

2.1.2 Web of Things and Semantic Web of Things

The premise behind the adoption of the *Web of Things* (WoT) is to leverage the widely popular web protocols, standards, and blueprints to make data and services offered by IoT Things more accessible to a larger pool of developer [34]. The WoT aims to effectively break the "Silos of Things" also known as "one device, one protocol, one app". To archive its goals, WoT is used in the application level to abstract the complexity and variety of lower levels (protocols, firmware, data formats) by using tools and available techniques on the Web technology. Thereby, it could facilitate the integration between IoT devices and applications. In other words, by hiding the heterogeneity of IoT devices behind Web technologies, the WoT allows developers to more focus on their solutions. In practice, the developers could interact with IoT Things via web browsers and explore the Things as surfing the web. The collected data from Things is visually displayed by using Web programming languages like *Hypertext Markup Language* (HTML), *Cascading Style Sheets* (CSS), and *Javascript.*

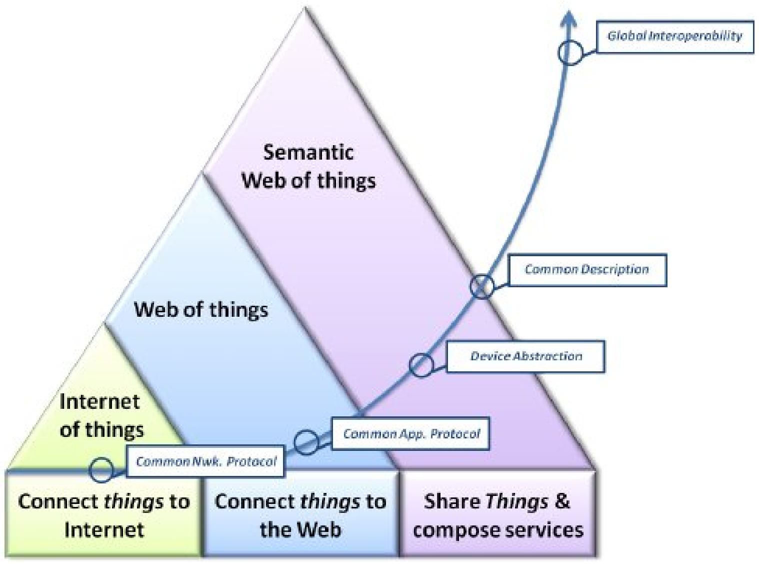


Figure 2.1 – The Evolution of Internet of Things solutions.

In general, the Web of Things facilitates the interaction and exchange of information between different Things and IoT systems powered by Web technologies. However, the exchanged data may be encoded and presented under different formats (envelopes, semantics, and meta-data). For example, the collected data representing current temperature can be encoded under the plain text, *Extensible Markup Language* (XML)/*Efficient XML Interchange* (EXI), or *JavaScript Object Notation* (JSON) format. The syntax can be "temperature" or "temp". Therefore, it is necessary to build a *Semantic Web of Things* (SWoT) to ensure a common understanding and format. In a more general view, the SWoT concept represents the evolution of the Web on IoT Things with the Semantic technologies. The goal of SWoT is to provide semantic interoperability that allows not only sharing and reusing the IoT Things but also making IoT data to be universally understandable [35]. The summary of the evolution from Internet of Things to Semantic Web of Things is illustrated in Figure 2.1 [36].

2.1.3 Massive Internet of Things

With the explosion of Internet of Things, the number of Things and its connectivity have been growing exponentially. In addition, the Things is not only deployed in the dense environments (e.g., building, city.), but also in the hostile environments (e.g., in the forest or mountain). As a result, *Massive IoT* (MIoT) is emerging as a new focal point for IoT connectivity technologies referring to the huge volume of constrained IoT devices, which stringently require excellent coverage, cost-effective and low-energy consumption [37]. Among several new connectivity technologies for MIoT, proprietary LPWAN technologies like Sigfox and LoRa have been considering the potential candidates while cellular-based connectives, such as *5G* or *Narrowband IoT* (NB-IoT), are under developing and testing process [38].

In practice, the IoT devices in Massive Internet of Things context are distributed in spacious areas from large manufacturing plants to inside sewer systems where the radio signal is physically challenging. To adapt to such environments, the connectivity technology of Massive IoT must be wide coverage and robust. In addition, replacing device battery in enormous areas is extremely expensive. They have to consume low energy to extend the device battery life. High throughput and latency are not unessential in massive IoT applications, since they more focus on collecting data than controlling the IoT Things.

2.2 Fundamentals of IoT

2.2.1 IoT Things

Similar to Internet of Things definition, deriving a unified definition for the "Things" in IoT is still challenging although it has received the most attention from academic organizations, such as *National Institute of Standards and Technology* (NIST), *Committed to Connecting the World* (ITU), *World Wide Web Consortium* (W3C), *International Energy Research Centre* (IERC), and *Internet Engineering Task Force* (IETF) [39]. *Institute of Electrical and Electronics Engineers* (IEEE) simply defined the Things as a physical or a relevant object from a user or application perspective [33]. However, IERC believes that a Thing can be a physical or virtual object and identified by an unique identity [40]. NIST proposes the Things can be software, hardware, or the combination of software and hardware. [41].

Due to the heterogeneity in Things definition, we simplify the Thing could be a physical or virtual object integrated into a network. This object can be interacted via a unique identity and interfaces. For example, a smartphone can be a Thing which is a physical object, able to connect networks (WiFi, Cellular Network), has a unique identity (phone number, IP address) and interfaces (Web services, applications).

2.2.2 IoT Connectivity

In this section, rather than mentioning all the IoT protocol following an existing architecture model like *Open Systems Interconnection model* (OSI model) , we only briefly present some dedicated protocols in IoT and categorize them based on their functionalities.

Infrastructure:

- *6LoWPAN:* This is an acronym of IPv6 over Low Power Wireless Personal Area Networks defined by IETF in the document RFC 6282 [42], deriving from the idea that "the Internet Protocol could and should be applied even to the smallest devices" [43]. This protocol uses 2.4 GHz frequency with 250 kbps rate.
- *uIP:* The uIP is an open source project licensed under a BDS style license [44]. The goal of this project is to create a dedicated TCP/IP stack for 8 or 16 bits micro-controllers. Currently, it is further developed by a wide community.
- *NanoIP:* The concept NanoIP is to optimize all features of Internet to adapt to embedded and small devices, without the overhead of TCP/IP [45]. NanoIP uses two dedicated transport techniques are nanoUPD and nanoTCP. A socket-compatible API is also provided to ensure the compatibility to the original IP protocol.
- *Time Synchronized Mesh Protocol (TSMP):* TSMP is a communication protocol designed for a self-organizing network of wireless devices enabling reliable, low power, and secure communication [46].

Discovery:

- *mDNS:* The mDNS is used to resolve hostnames to IP addresses within small networks. Except missing a local name server, this technology is essentially the same with the unicast Domain Name System (DNS) in term of programming interfaces, packet formats, and operations [47].
- *Physical Web:* The Physical Web aims to discover and interact with nearby devices through a list of URLs being broadcast. That means every smart object in the network needs to broadcast its access URL to nearby devices.
- *HyperCat:* HyperCat is an open, lightweight JSON-based hypermedia catalogue format to exploit Thing resources [48]. It allows adding a set of semantic annotations to Things resources and makes them discoverable over the web.
- *Universal Plug and Play (UPnP):* The UPnP uses Internet and Web protocols to automatically discover new devices to be plugged into a network. These devices announce their presence to other devices by using a discovery protocol based on Hypertext Transfer Protocol (HTTP) [49].

Data Protocol:

- *Message Queuing Telemetry Transport (MQTT):* MQTT is a publish-subscribe based messaging protocol working on the top of TCP/IP protocol. It is effective for connections limited bandwidth [50]. An MQTT system consists of a central messaging server named "message broker" and clients. There are two client types are (1) Publisher: Clients publish data to broker. (2) Subscriber: Clients receive data from the broker. The responsibility of brokers forward data from publishers to subscribers.
- *Constrained Application Protocol (CoAP):* CoAP is an application layer protocol designed for constrained internet devices limited in storage, computation power. It is based on RESTful protocol to directly translate to HTTP for simplified integration, and while also fulfill specialized requirements, such as multicast

support, very low overhead, and simplicity [51]. Currently, the major standardization for CoAP is done by IETF, and various new functionalities have been added [52].

- *Extensible Messaging and Presence Protocol (XMPP):* XMPP is a real-time communication protocol based on Extensible Markup Language (XML). It is defined in an open standard managed by IETF. Designed to be extensible, this protocol is also used for the publish-subscribe model in Voice over Internet Protocol (VoIP), video, and IoT applications.
- *Advanced Message Queuing Protocol (AMQP):* Similar to XMPP, AMQP is an open standard application layer, but it is designed for message-oriented middleware. Thereby, its functionalities ensure reliability and security, such as message orientation, queuing, routing [53]. The authentication and encryption are based on Simple Authentication and Security Layer(SASL) or Transport Layer Security (TLS).
- *Lightweight Local Automation Protocol (LLAP):* LLAP is a simple short message designed for the device-to-device communication. The key strengths of LLAP are widely compatible and easily understandable by humans [54].

Communication/Transport layer:

- *Sigfox:* Sigfox is a radio technology using Ultra Narrow Band (UNB). It targets to provide a long range and low energy consumption connectivity for IoT devices. By using UNB, sigfox achieves bandwidth efficiently, low noise levels. However, it is limited in throughput (100bps) and packet size (12 bytes payload).
- *Narrow band IoT(NB-IoT):* NB-IOT is a Low Power Wide Area Network (LPWAN) radio technology developed by 3rd Generation Partnership Project (3GPP) [55]. It adopts narrowband technology with single frequency 200kHz. Thereby, NB-IoT has the limited bandwidth. In contrast with Sigfox, NB-IoT aims to provide the lost-cost connection in indoor scenarios or dense urban areas in which the connection density is high.
- *LoRa:* LoRa is a patented digital wireless data communication IoT technology developed by Semtech [56]. It operates over the open license radio frequency bands (169 MHz, 433 MHz, or 868 MHz) enabling long-range transmissions with low power consumption [57]. The technology of LoRa includes two parts: (1) LoRaPhy: is a communication technology working on the physical layer to enable long-range communication link; (2) LoraWan: is an open-source communication protocol built upon the LoRaPhy.

2.2.3 IoT Platform

The IoT platform (also known as IoT middleware) is an intermediate software layer interposed between the technique and application layer. Its goal is to abstract the IoT system complexities related to hardware, connectivity, configurations under simple interfaces or services. In this way, the application developer more focusing on their tasks without concerning the technology behind. In the IoT, such complexities mostly relate to communication because of the considerable heterogeneity in devices, protocols, and applications. In such context, the middleware provides standard services or interfaces, which wrap the heterogeneous computing and communication technologies of IoT devices. To achieve its goals, the middleware needs fulfilling requirements [58]:

- *Programming abstraction:* The middleware should provide a simple and unified API interface for application developers. This interface is used to separate the IoT applications and services with the underlying heterogeneous IoT infrastructures. The style of the programming interface depends on the interface type. For instance, SQL-like languages for data query will be used in descriptive interfaces. [59].

- *Interoperable layers:* One of the main goals of middleware is to deal with the heterogeneity in IoT. Hence, the middlewares should equip an interoperable layer interacted with heterogeneous components in IoT systems.

- *Service-based:* A middleware should equip elements to produce high flexibility and reliability services which easily adapt to the frequent changes in middleware functions and application contexts.

- *Adaptive:* A middleware must be able to dynamically adapt and adjust itself with the changes of its context or environments.

- *Context-aware:* Context-aware represents the ability to aware the context of users, devices, and environment. This is a key feature to build an adaptive middleware.

- *Autonomous:* The middleware may integrate, communicate and exchange the data with IoT devices without human interactions. To achieve this requirement, there are many technologies including *autonomous agents*, *embedded intelligence*, and *proactive approaches* [60, 61].

2.2.4 IoT Service

As IoT service is an ambiguous term which highly depends on the context, it is hard to provide a concise definition. The most common understanding is that "An IoT-Service is a transaction between service providers and consumers. It triggers a prescribed function to interact (e.g., observe, initiate actions) with the physical world via IoT Things" [62]. Based on this definition, we could classify the IoT services into two types:

- *Thing-based services:* This kind of services provided by the IoT Things allows to connect, obtain and control the Things resources. It is popular in powerful IoT things, such as smartphones, Raspberry devices.

- *Cloud-based services:* These services are provided by the cloud-based IoT platforms. They may include Things-based services and non-IoT services.

2.2.5 IoT Applications

The IoT is expected to offer a huge number of applications, which significantly increase our life quality in every aspect (e.g., working, living, traveling) . In this section, we briefly present the common IoT applications in: (1) Transportation domain, (2) Health care domain, (3) Smart environment domain.

- *Transportation domain:* Vehicles will be more intelligent based on collected information from surrounding context, such as traffic status, road quality, and nearby objects. Two typical features in vehicle based on IoT technologies are *Collision avoidance system* and *Automotive navigation system.* In detail, collected information is used by a collision avoidance system to assist drivers in preventing unexpected objects (e.g., big stones, other vehicles) on the road. The information related to traffic jam or incidents is used to offer the optimum paths to drivers. Regarding the logistic domain, the condition of transported goods is monitored in real-time and sent to a management system. Based on this information, the managers manage their product quality and optimize the supply chains.

- *Health care domain:* Wearable devices are equipped sensors to monitor patent conditions in real-time. Based on the collected information, doctors could diagnose potential diseases and problems. Moreover, identifying medications and patients reduce serious incidents in treatment processes (such as the wrong drug, time, procedure). For example, in a smart hospital scenario, all medications are tagged by RFID tags with detail information about patients and usage instructions. In this way, they are simply and correctly checked before delivering to the patient. Therefore, harmful incidents caused by wrong drugs, treatment procedures are reduced.

- *Smart environment:* A smart environment makes us more comfortable and convenient based on the intelligence of surrounding objects. For instance, sensors and actuators in our office or house could automatically adapt temperature or light following the time or our preferences. Moreover, smart environments improve the automation in industrial plants by deploying a large number of sensors and Radio-frequency identification (RFID) tags. For example, instead of manually checking the origin and conditions of products, workers scan RFID tags to gain all the necessary information.

2.3 IoT Cloud

2.3.1 Cloud Computing

Following the definition provided by NIST, cloud computing is a model for enabling ubiquitous, convenient, on-demand access to shared pools of configurable computing resources (networks, servers, storage, applications, and services) [63]. Over last decade, it has been strongly impacting the IT industry by offering virtually unlimited storage and computing power at low cost [64]. Based on these advantages, the IT companies could quickly implement and deploy complex IoT solutions. Global companies (Amazon, Google, and Facebook) are typical examples of gaining enormous benefits by widely adopting cloud computing. Figure 2.2 presents the major aspects of Cloud including essential characteristics, layered architecture, and standard services mode [65]. In general, cloud computing architecture is divided into four layers:

- *The hardware layer:* This layer is used to manage physical resources, such as physical servers, routers, switches. In practice, it is deployed in a data center.

- *The infrastructure layer:* This layer uses virtualization technologies to perform the partition of physical resources into virtual pools (storage or computing).

- *The platform layer:* This layer is used to directly distribute applications to virtual resources. It consists of operating systems and application frameworks.

- *The application layer:* At the top of architecture, the application layer contains various cloud applications that are better performance, availability and lower operating cost in comparison with traditional applications.

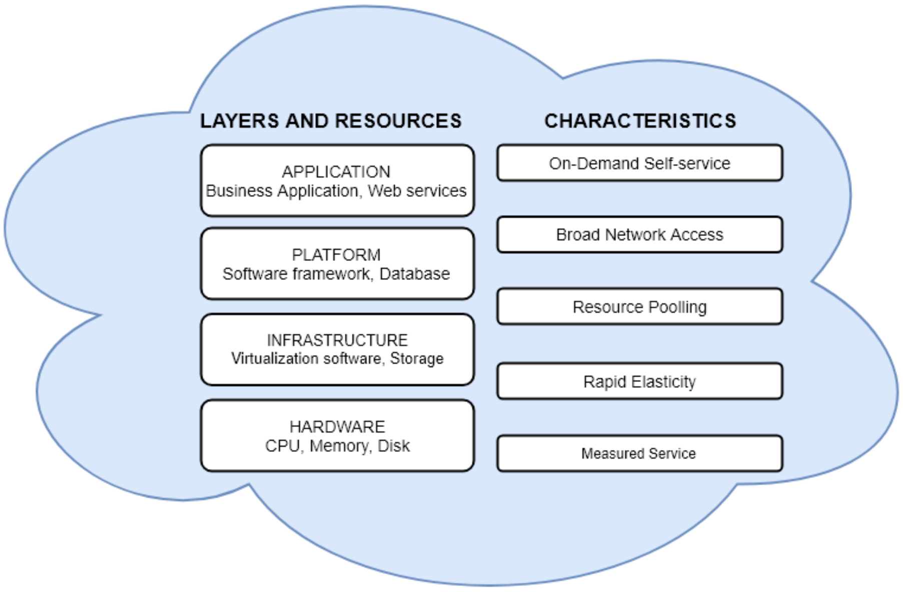


Figure 2.2 – The cloud computing overview.

2.3.2 Cloud-based IoT

IoT and Cloud computing are disruptive technologies in the Internet era. Fortunately, their characteristics are often complementary in many aspects. Thus, a novel IT paradigm, namely Cloud-based IoT or CloudIoT, is born from the idea about integrating IoT and Cloud Computing [66, 67]. The primary benefit of such integration fall in three categories:

- *Communication:* Cloud provides effective solutions to connect, track, and manage the Things (e.g., devices, actuators) from anywhere at any time through cloud applications or portals. In addition, the high-speed networks allow access and manage in real-time the IoT data stored on cloud [68].
- *Storage:* With billions of collected devices, the IoT produces an enormous amount of non-structured or semi-structured data [15] which also possesses typical characteristics of big data: high volume (data size), high variety (data types), high velocity (data frequency). In such scenarios, cloud computing is the low-cost and effective solution based on large-scale and long-lived storage capabilities. Moreover, storing data on cloud makes the data more secure and in the way that is easily accessible.
- *Computation:* The particular properties of IoT Things are restricted computation power and energy so that they cannot perform complex operations or data processing. Hence, most of the collected IoT data is raw and only processed after conveying to more powerful nodes (gateways, brokers, and routers). But these nodes can not handle a large number of concurrent connections and their hardware is not dedicated to data processing. In this way, achieving scalability are very challenging. To deal with these challenges in IoT, cloud offers enormous computing power along with on-demand usage model. The cloud resources including several dedicate servers can be elastically

provisioned and released to respond rapidly to demands. These capabilities enable IoT data processed on-the-fly, scalability, and managing complex data processing model [13, 69].

2.4 Interoperability in IoT

2.4.1 Interoperability Definition

In the IoT context, there is no unique definition of the *Interoperability* term. The most common understanding within the community is that: "Interoperability is the ability of two systems to interoperate using the same communication protocol". In a more general view, IEEE defines interoperability as "the ability of two or more systems or components to exchange information and use the exchanged information" [70]. This ability could be separated into four levels as illustrated in Fig. 2.3 [71].

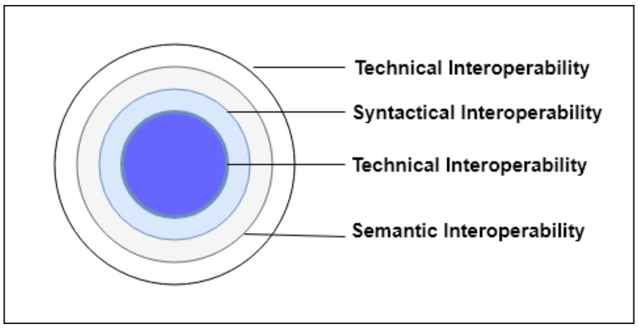

Technical Interoperability
Syntactical Interoperability
Technical Interoperability
Semantic Interoperability

Figure 2.3 – The levels of interoperability.

- *Technical Interoperability:* This is the lowest interoperability level, usually associated with hardware or software components, systems, and platforms enabling machine-to-machine communications. In other words, technical interoperability represents the ability of IoT components to "talk" with each other. Hence, it focuses on communication protocols and network infrastructures.
- *Syntactical Interoperability:* After successfully communicating, the next level of interoperability is related to the data formats which are transferred by communication protocols. Syntactical Interoperability represents how machines could understand the exchange information.
- *Semantic Interoperability:* This level is typically related to the human interpretation of the content. In more detail, Semantic Interoperability represents the mutual understanding between peoples about the collected data meaning.
- *Organizational Interoperability:* It is the highest level of interoperability in which diverse organizations could effectively communicate and transfer information regardless of their systems or infrastructures. The organizational interoperability is reached if three previous levels are fulfilled.

2.4.2 Interoperability Taxonomy

To understand the IoT interoperability in more detail, we analyze it from different perspectives. The interoperability is classified into device interoperability, networking interoperability, syntactic interoperability, semantic interoperability, and platform interoperability [72].

- *Device interoperability:* The device types in IoT are highly diverse. But, there are two major device types: (1) *The high-end devices* possess sufficient resources and computational capabilities, such as Raspberry Pi, smartphone; (2) *The low-end device* also known as constrained devices are limited energy, computational power, and communication capabilities, such as RFID tags, tiny sensor, and actuator [73]. Moreover, many communication protocols have emerged because of the complex requirements of IoT solutions, such as Lora, Sigfox, and NB-IoT. In the missing of a global communication standard, one IoT device is impossible to equip all communication methods. However, in practical, IoT devices have to exchange information with different device types using different communication methods. In such scenario, we need to have interoperability between heterogeneous devices. In summary, device interoperability involves two aspects: (1) the exchanged data between heterogeneous IoT devices using heterogeneous communication methods; (2) the ability to integrate new devices into any IoT platform.

- *Network Interoperability:* The interoperability at network level aims to seamlessly exchange information through different networks by using end-to-end communication methods. To achieve these goals, IoT network systems have to deal with the issues related to addressing, routing, resource optimization, security, and mobility [74].

- *Syntactical Interoperability:* The data exchanged between heterogeneous IoT systems is usually packed and encoded under different formats [75]. As a result, the message receivers cannot correctly decode and obtain the content of such messages. Syntactical Interoperability aims to interoperate the format as well as structure of messages exchanged between heterogeneous IoT Things and systems.

- *Semantic Interoperability:* Following W3C, semantic interoperability is the ability to enable numerous agents, services, and applications to exchange information, data, and knowledge in a meaningful way [76]. In IoT context, the exchanged data usually use different data models or schemas. This leads to the semantic incompatibility in exchanged information between IoT systems, even if these systems have presented their data and resources to others [77].

- *Platform interoperability:* The IoT Platform interoperability is the ability to enable interoperability across IoT platforms in various domains. The Figure 2.4 shows the general concept of platform interoperability. The heterogeneity of IoT platform providers is the main cause of platform interoperability issues. For example, Apple Homekit supports only *Swift programming language*, AWS IoT offers an SDK for embedded C and NodeJS. Due to such non-uniformity, application developers need to intensively understand API, information model of each platform to develop IoT applications.

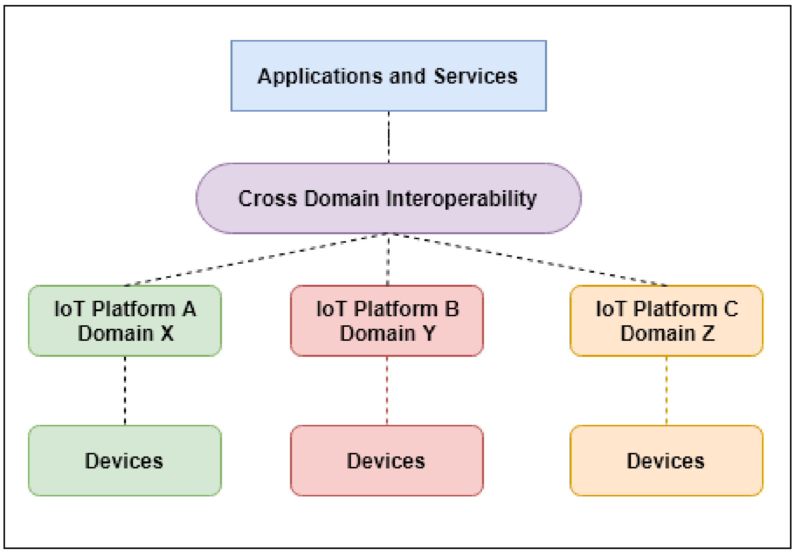


Figure 2.4 – The platform interoperability.

2.5 Data Outliers in IoT

2.5.1 Data Outliers Definition

In practice, collected data from IoT devices (e.g. sensors, smartphone) is typically unreliable due to the presence of "dirty-data" also called *outlier* or *anomaly*. These data outliers are caused by:

- *Dropped readings:* The IoT Things are often constrained devices deployed in a restricted environment (e.g., in sewer network, giant factory, plans). Therefore, the collected data is usually intermittent due to communication errors and scarce resources.
- *Unreliable readings:* Sensor errors, calibration failure among other reasons lead to abnormal data.

The most common definition is that outliers are outside what is considered as a "normal state" [78]. Depend on the context of the data, we determine the normal state. For example, 40 Celsius degree in summer is normal but it is considered as abnormality when reported in winter. In [22], outliers are considered "events with extremely small probabilities of occurrence". In another way, they are also seen as "points in a data set that is highly unlikely to occur given a model of the data." [79] .

2.5.2 Data Outliers Taxonomy

Based on discussed definitions, outliers are significantly different from the rest in a dataset. It does not mean all outliers present the errors. In some cases, the outliers contain essential information about the changes of the sensed environment. For example, the notably increase of soil humility presents the raining or watering events. Thus, an outlier could be an error or an event [80].

- *Erorrs:* An data error involves in a noise measurement or data coming from a faulty IoT device. In practice, the outliers caused by errors significantly outnumber the one caused by events. As such errors reduce the data quality, they need to be correctly identified. Depend on the applications, these errors could be eliminated or repaired.

- *Events:* An event refers to the particular phenomena reflecting the changes of the real world (forest fire, watering, raining.). This kind of outlier occurs for a long period and creates a particular pattern in dataset. However, faulty devices may also create such behaviors. Therefore, it is hard to distinguish between events and errors by simply examining dataset statistic.

The outliers caused by errors is classified into three groups [81]:

- **Point anomalies**: A single instance of data is anomalous if it is significantly different from the remaining data.
- **Contextual anomalies:** The abnormality is context specific. This type of anomaly is common in time-series data. For example, 30 Celsius degree during summer is normal but may be abnormal in winter.
- **Collective anomalies:** This anomaly type contains a set of consecutive point anomalies. This pattern does not comply with the dataset distribution.

2.5.3 Data Outliers Detection

Outlier detection is a process of discovering the elements that significantly differ from what is considered as normal [22]. The final goal is to both eliminate outliers caused by errors and highlight ones caused by events. The outlier detection is an important step in the data cleaning process to increase data quality. In addition, highlighting outliers caused by events could reveal rare events and patterns underlying in a dataset. The concept of outlier detection is closely related, but much broader than noise removal. It also close to novelty detection which targets to identify the novel pattern in dataset [81]. Outlier detection method is cataloged into supervised, unsupervised and semi-supervised methods depending on prior information requirements.

- *Unsupervised Outlier detection:* This technique does not require labeled data. The outliers are detected under the assumption that the majority of dataset is normal and abnormal data is only a small proportion.
- *Supervised Outlier detection:* This technique requires a training dataset in which normal and abnormal states are labeled. Such dataset is used to train a classifier.
- *Semi-supervised Outlier detection:* This technique constructs a classifier model for normal state from clean datasets which are missing abnormal data. Then, this model is used to detect abnormal data.

2.6 Energy-efficiency in IoT Devices

2.6.1 Energy-efficiency Definition

The *energy-efficiency* concept was first presented as the proportion between the total amount of delivered data and total consumed energy [82]. This means the value of energy-efficiency is increase if more data is transmitted with less energy consumption. However, energy-efficiency described in a broader view is “using less energy to provide the same service quality”. For instance, a system provides a higher prediction quality while reducing energy consumption could be considered as energy-efficient.

The IoT generally consists of small devices with limited power and battery capacities deployed over a wide geographical area. In addition, recharging or replacing battery could be costly or even impossible because these devices are usually deployed in hostile or restricted environment (e.g., underwater or ground). In many cases, battery life may be required several months or even years. Therefore, the energy-efficiency mechanisms have been gaining a vast of attention in IoT recently.

2.6.2 Device Energy Consumption

To produce an energy-efficiency method, we need to deeply understand the energy consumption model of IoT devices. The authors in [83] examined three main sources of power consumption including communication, computation and sensing operation. Although different device types have different energy consumption profiles, the overall remarks are:

- The communication operations consume much more energy than the computation operations. Therefore, we can trade communication for computation.
- The radio energy consumption for the reception, transmission, and idle states are the same while the power consumption in the sleep state is significantly low. Therefore, the radio should be turned off whenever possible.
- The sensing operations also consume high energy, so it needs to be reduced as much as possible.

These observations are verified at a sensor platform named TelosB. The authors measured this platform in 95 hours with 100% duty cycle (no sleep, radio always on, no communication), and 200 hours with a 25% duty cycle. As a result shown, the radio module is the most energy-consuming component. In addition, we can save energy about ten times in the idle state in comparison with receiving data.

3 Related Work and Challenges

3.1 Interoperability in IoT

3.1.1 Related Work

To enhance the interoperability in IoT, researchers have leveraged several approaches and technologies from other fields, such as semantic web, cloud computing, wireless sensor network, and fog computing into IoT. In this section, we present an overview of existing approaches as well as their challenges to achieve interoperability. For each proposal, we focus on its interoperability levels (technical, syntactical, semantic and organizational interoperability) and interoperability types.

3.1.1.1 Adapters/gateways

Gateways or adapters are the class of schemes aiming to improve the device interoperability. This approach uses an intermediate tool called *mediator* or *connector* installed in IoT gateways or adapters to converse protocols, data, and standards between sending and receiving devices. For instance, an IoT device using Bluetooth connectivity desires connecting to other devices using ZigBee connectivity. In this case, the gateway is installed dedicated hardware or software used to convert from Bluetooth to ZigBee and, vice versa.

The most critical challenge of this approach is scalability. With a large number of heterogeneous IoT devices, it demands considerable efforts to implement and deploy connectors on gateways. Also, the converting performance must be considered. For example, if a system needs to support n connection types, we have to develop $\frac{n*(n-1)}{2}$ connectors. This number of connectors cannot be implemented in a single gateway. Therefore, several *one-to-many protocol gateway* solutions may be used [84].

Several works in both academic and industry related to design and standardize IoT gateway have been proposed. Ponte [85] presents a framework enabling data exchange between various IoT devices through different connectors. The main limitation of this framework is that it supports a few protocols, such as HTTP, CoAP, and MQTT. In addition, developing new connectors is complex and requires programming skills. Zhu et al. [86] propose an IoT gateway architecture using *user-space programmable software* to interoperate between wireless sensor network (WSN) protocols and mobile communication networks or Internet. This gateway supports data forwarding, protocol conversion, and management. However, accessing collected data via simple API is unsupported. Using the same method, the authors in [87] present a gateway architecture adapting to the differences in device protocols and security issues. But, its scalability is not mentioned. Leveraging the computation power of smartphone, the authors in [88] and [89] build a mobile gateway supporting the same functions with the IoT gateway. Its limitation is heavy energy consumption. The *Semantic Gateway as a Service* (SGS) is presented in [90]. It is an IoT gateway providing the semantic interoperability for IoT system. The raw sensor data is transmitted to center gateways via proxy layers that support multi-protocols. Then, it is added semantic annotations defined by a *sensor network ontology.* This step provides the semantic interoperability for collected data. However, the scalability and energy consumption of this approach are very limited.

3.1.1.2 Virtual networks

The first concept of *Virtual Network* is presented in [91] aiming to integrate *Wireless Sensor Network* to the Internet. It is built on the top of physical network enabling end-to-end communication using different protocols. In this way, application developers could interact with devices, sensor, and actuators in the physical network via functions provided by the virtual network. This concept is used to develop *Internet of Things Virtual Network* (IoT-VN) [92]. The device interoperability is achieved by integrating all heterogeneous IoT devices into the same virtual network. However, in practice, this integration is impossible due to the fragment of IoT markets and the enormous number of IoT devices. Furthermore, the scalability of virtual network in large-scale deployment is still challenging.

3.1.1.3 Networking technologies

Several networking protocols and technologies have been proposed to ensure the interoperability in the IoT networking layer. In this section, we present the existing solutions to achieve this goal.

Software-defined networking (SDN): SDN is a new networking paradigm enabling efficient network configurations by separating the forwarding process of network packets from the routing process. Thereby, the current wireless and mobile networks based on SDN are more intelligent, efficient, and secure [93]. based on these advantages, Matinez and Skarmeta [94] use SDN to enable communication between IoT devices using IPv6. They add an IoT controller over SDN controller to facilitate the device management operations. Thus, even if the devices have different protocols, this controller could convert it into unified ones. In [95], the authors provide a middleware equipped an *IoT SDN controller* to manage the heterogeneity in IoT multi-networks. They use a central controller for monitoring and coordinating the existing devices and data flows in the system. However, the current SDN technologies do not support *traditional networking devices* (such as routers, gateways which have old firmware version) [96]. Therefore, we need to have novel solutions to abstract the traditional networking devices in SDN regardless of their specific hardware configurations [97].

Network function virtualization (NFV): The network function virtualization is a complementary approach of SDN. It separates the physical hardware of the network from the software layer running on them. Thereby, the NFV enables creating several services on the same physical hardware layer. The authors in [98] introduce a general IoT framework by combining IoT architecture with SDN architecture and NFV. The framework consists of (1) APIs layer for developing IoT application; (2) middle layer containing distributed network OS created by NFV; (3) lowest layer containing SDN switches and IoT gateway. However, the NFV limitations are the complexity and security issues in maintaining the network OS. In addition, virtualization has many challenges about resource management, complex operations, and security [99].

Fog computing: The integration between Internet of Things, Cloud computing, and Fog computing arises a novel concept named *Fog of Things*, where the computing, storage, and networking services are placed at the edge devices (such as routers, gateways, routing switches, and network stations) rather than centralized cloud servers [100]. In this way, the raw data collected from IoT devices is converted into usable knowledge and added semantic annotations before being available on the Web. This increases data interoperability in IoT and reduces the network latency [101]. The authors in [102] combine fog computing and smart gateway to propose an "IoT Hub" that supports typical server services (e.g., resource discovery, caching, reprocessing and trimming the collected data). This concept is extended in [103] to manage the heterogeneity in IoT Things. To quickly react to the changes of sensing environment, [104] introduces an edge computing architecture using Virtual IoT device.

3.1.1.4 Open API

API is an interface written by a programming language. It is used to access data or functions of applications. Thus, to enable interoperability, the API must be well-documented and open for all developers. Most of current IoT platforms provide a public API based on RESTful principles, and allow common operations (such as PUT, GET, PUSH, or DELETE) to support developers access their services. However, these APIs are designed as platform-specific or proprietary [105]. In detail, the API syntax, endpoint and, returned data are self-defined by service providers regardless of standards. This leads to the heterogeneity in the syntax and API operations. For example, an IoT application is used to control the air conditioner. This application could increase temperature via an API provided by the air conditioner provider. If the application desires to control air conditioner of other providers, without standard API, the developer must write new dedicated codes for new providers. However, with a standard API, the application only changes the end-point (the new air conditioner address). To bridge this gap, several approaches have been proposed. HyperCat provides a specification enabling syntactic interoperability between different APIs and services, which are described under Catalog formats [106]. The resources in a catalog are identified by URI and tagged with metadata. The Big-IoT European projects have been working on a generic *inter-working API* which allows accessing to resources of all existing IoT platforms. This API acts as an adapter and needs to be implemented by other platforms to achieve interoperability [107].

3.1.1.5 Service oriented architecture (SOA)

To enable device and cross-platform interoperability, the researchers add *Service Oriented Architecture* layer on the top of network layer so that the devices and collected data are effectively managed via *service components* [108, 109]. In detail, the functions and operations of devices are wrapped by these components. Thereby, IoT applications simply expose the device resources via provided service APIs. As a result, the network and device interoperability are significantly increase. The authors in [110] apply web service technologies to SOA to maximize the device and data interoperability. [111] using classic web service-oriented approach (WS-* web service) and [112] using resource-oriented approach (REST web services) aim to increase the syntactic interoperability. Pautasso et al [113] compare the benefits of WS-* web services and REST web services to SOA in various use-cases. They claim that REST web services are suitable for tactical integration over the Web, whereas WS-* web services fit for enterprise applications.

3.1.1.6 Semantic web technologies

The *Semantic Web technologies* are designed to describe web resources semantically. Currently, many research directions leverage the benefits of Semantic Web technologies into IoT to achieve the semantic interoperability. The typical paradigm of such integration is the Semantic Web of Things targeting to mutually understand the IoT data and entities (such as services, devices) between people [114] by using shared standards, vocabularies in a schema or an ontology.

On the Semantic Web of Things, *Ontologies* define the concept and relationships of terms in a domain [115]. The concept of ontology can be an object (such as person, car, building) and the relationships is the relation of two concept. The ontology is used to prevents the ambiguity or heterogeneity of the terms between different domains [115]. Many ontologies for IoT have been proposed, such as W3C Semantic Sensor Network (SSN) [116], SAREF [117], and OpenIoT [118]. A study of the existing ontologies in several domains is presented in [119]. They also describe in detail how to use ontologies to achieve cross-platform interoperability. However, there is no global ontological standards. Most of the existing ontologies are domain-specific [119].

Several IoT research projects leverage the benefits of ontologies and other semantic technologies to enhance the interoperability in IoT. *Semantic Sensor Web* (SSW) [120] is the adoption of Sensor Web and Semantic Web technology. SensorML [121] provide by Open Geospatial Consortium (OGC) is XML-based standard to describe sensor of web. UbiROAD [122] introduces a framework enabling the semantic interoperability at data and functional protocol level. Serrano [123] analyzes the current semantic interoperability challenges in IoT. Base on this analysis, the authors provide a methodology, namely SEG, to achieve semantic interoperability at the application layer. They also add the semantic annotations to heterogeneous IoT data to assist developers in building IoT applications. In another way, the authors of [124] present a set of semantic models for describing IoT components. In addition, they introduce a novel concept named *sensing as a service* supporting access to IoT resources and functions through standard services.

3.1.2 Open Challenges

Although many IoT solutions (standards, platforms) have been proposed to achieve interoperability, there are existing challenges in this topic. In this section, we will present major challenges in IoT interoperability based on reviewed solutions.

- *Large-scale device integration:* The IoT device is a critical element in IoT system. Thus, interoperability for devices is vital for the success of IoT. Although several methods have been proposed, integrating heterogeneous devices into an IoT platform in large-scale (large device volume and various data types) is still a major challenge. Most of reviewed approaches are limited to the scalability and flexibility to deal with the rapid growth of IoT devices in both quantity and type. Furthermore, direct communication between two heterogeneous devices is still a unresolved issue in IoT.
- *Cross-platform interoperability:* The reviewed IoT platforms provide an open APIs to access their services. However, these APIs are designed from custom RESTful principles and data models. In addition, the integration should not require the major changes in the platform architecture, even if they have differences in technologies, underlying features and provided services. Thereby, cross-platform interoperability between these platforms is very challenging.

- *IoT data interoperability:* Various reviewed methods provide the standards and semantic ontologies to address the interoperability in IoT data. However, it does not mean that these proposals are globally accepted and used. Moreover, most of them are domain-specific. It is hard or even impossible to integrate all existing standards into a consistent, coherent one. In addition, the new device types along with their own data format, which are uncompliant with existing standards, is emerging everyday.Therefore, collecting and processing data from heterogeneous data sources to maximize usable knowledge is still an open challenge in IoT.

3.2 Reliability in IoT

3.2.1 Data Reliability

3.2.1.1 Related Work

In IoT context, most of the collected data from IoT Things is uncertain and inconsistent. Thus, along with data acquisition and data mining, data cleaning is a crucial step to gain the success of IoT paradigms or services. It also helps the IoT application developers more focusing on core solutions than prior processes to ensure data reliability. In general, data cleaning consists of two main steps: (1) Outlier detection: identifying the errors or events in data; (2) Data repairing: repairing the identified errors. Several methods in both academic and industry have been proposed. In this section, we briefly review common data cleaning approaches.

3.2.1.1.1 Outlier detection

Unsupervised methods: One of the most common unsupervised outlier detection approaches is the concept of nearest neighbor analysis. Such techniques detect the abnormality by examining the distance or similarity between two data instances. Normal data instances occur in dense neighborhoods, while anomalies occur far from their closet neighbors [81]. The distances (or similarities) are calculated differently. For example: *Euclidean distance* is the wide usage for continuous attributes in [125, 126, 127]. For categorical attributes, a simple matching coefficient is often used, but more complex distance measures is also used in [128, 129]. The original idea of *nearest neighbor anomaly detection* defines the anomaly score of a data instance as its distance to its k^{th} nearest neighbor in a given data set which is mentioned in [130]. This work also has been applied to detect shorted turns in wind turbine-generators in [131]. The basic technique is enhanced by researchers in various aspects. For example: The authors in [132, 133, 134] calculate anomaly scores from the sum of the distances to k nearest neighbors. [135] counts the number of nearest neighbor within d distances as anomaly scores. [136] designs a simple sampling technique to reduce the complexity of the algorithm. A *Resolution Outlier Factor* (ROF) is proposed in [137]. According to this method, points are outliers or within a cluster depends on the resolution of applied distance thresholds. Another approach is based on the idea: "An instance that lies in a neighborhood with low density is declared to be anomalous while an instance that lies in a dense neighborhood is declared to be normal". The most well-known algorithm in such technique is *Local Outlier Factor* (LOF)[125]. For a given data instance, the anomaly score is the ratio of the average local densities of its k-nearest neighbors over the local density itself. However, LOF ineffectively detects the regions which are not separated. Several works were subsequently proposed to extend the concept of LOF. The authors in [138] uses the symmetric nearest neighbor relationship to define the outlier score. [139] introduces a new variation of LOF named *Connectivity-based Outlier Factor* (COF)[126] which can detect the abnormality distributed on arbitrarily shaped clusters. The LOF is also combined with other techniques. For example, the work in [140, 141] calculate the anomaly score, named *Cluster-Based Local Outlier Factor* (CBLOF), from local distances to nearby cluster and the size of the clusters. A more recent proposal is presented in [142] using relative entropy as the distance measurement.

Supervised methods: Supervised anomaly detection methods require the training data in which correctly labels both normal and abnormal data. Several approaches have been proposed, such as MetaCost [143] uses a relabeling approach to classification. The general idea of this method is to relabel some data points in training dataset by using *a Cost* so that normal data points have a reasonable probability to be classified into abnormal data. This work aims to make the training dataset more balanced. Support vector learning has been applied for outlier detection, such as one-class learning [144] and support vector data description [145]. The premise idea is to learn a boundary that encloses the normal data so that all outside data points are considered as outliers. Other approaches use the transitional machine learning classifiers to classify the dataset into normal and abnormal classes, such as the *Bayes classifier* [146], *nearest-neighbor classifier* [147], *decision trees* [148, 149], *rule-based classifiers* [150, 151] and *SVM classifiers* [152, 153].

3.2.1.1.2 Data repairing

Smoothing-based Data repairing is a light-weight and simple technique used for online data repairing. A typical example of this approach is the *simple moving average* (SMA) [154] calculated the values of current points from the mean of the last k points. Instead of using unweighted mean, the *exponentially weighted moving average* (EWMA) [155] multiples this mean with a weight value, which is exponential decrease over the time. Other approach named SWAB smoothing [156] uses a regression function to repair streaming data. Based on last sliding windows, SWAB approximates the data trend by a linear interpolation function. Despite the lightweight and simple implementation, these smoothing methods have very low repair accuracy. In addition, they also change the originally normal points in the repairing process.

Constraint-based data repairing techniques repairs data based on given constraints, while minimizing the repair modification [157, 158]. The SCREEN algorithm [159] works under the assumption that the speed of data changes is constrained. This means the *jump* of values is limited and considered as an anomaly if it is out of a given boundary. Based on such assumption, they propose a solution for stream data to identify and repair the "jump" values in a given sequence (windows data) w.r.t the speed constraint while minimizing the repair distance. To achieve this goal, the authors introduce a novel concept named *Median Principle* to find the middle point of a specific sequence that is intuitively believed minimizing the repair distance. In addition, to deal with the tradeoff between choosing the window size and speed constraints. They proposed an adaptive windows size based recently extreme speed constraint (min or max speed). However, the SCREEN can show high performance in repairing single anomaly but hardly handle a collective anomaly. Moreover, the repairing results strongly rely on the correctness of the initial assumption. Being aware of such limitations, a latter algorithm named *Iterative Minimum Repairing* (IMR) [160] is proposed. The general idea of such algorithm is that combining between labeling some dirty observations and iterative repairing from high to low confidence repairs could enhance the performance. After each iteration, the parameter of the *Autoregressive with exogenous* *input* (ARX) [161] model is calculated to generate the repairing candidate list based on the difference between original and inferred data. The data point with the minimum difference is repaired. The stop conditions of repairing procedure could be reaching the threshold of convergence or the maximum number of iterations.

3.2.1.2 Open Challenges

Ensuring the data quality, especially in the IoT context, has faced many challenges. Most of the solutions are dedicated to specific purposes, such as uniquely detect anomaly or clean data. There is no comprehensive solution from identifying to repairing errors. Apart from that, current data cleaning solutions are still lack of:

- *Scalability:* With the exponential growth of IoT, the collected data from IoT devices is massive. The advantages of unsupervised method are light-weight and simple. However, their accuracy is very low. In contrast, supervised approaches could offer high accuracy, but they require a vast amount of time to train the model. Therefore, we need a solution that ensures scalability while maintaining high accuracy.

- *Flexibility:* IoT data is collected from various data sources (e.g., sensors, devices, RFID tags). The data cleaning techniques should analyze and combine the relations of this data to increase accuracy. In addition, the proposed techniques have to handle different variables, which present the end-user interests. Depending on the user-cases, the user may require different detection qualities. For example, fleet management application requires high accuracy in discriminating the sensor errors and events whereas applications monitoring CO2 or temperature do not require such accuracy level. This requirement also leads to a new challenge on how to map between algorithm outputs and the user's desired quality.

- *Distinguish error and event:* The primary feature missing in all data cleaning techniques is the ability to distinguish outliers caused by errors from those caused by events. Currently, data cleaning technique is often applied to filter out dirty data. This means the detected points are simply discarded as useless noises. Unfortunately, the eliminated data may contain notable events. These events occur by accident (e.g., a fire in a forest) or because of human intervention (e.g., watering the tree). Preserving these events is essential to interpret the context. For example, to optimize the watering schedule, a city environment management company deploys the sensors to monitor the impact of watering on soil humility. However, a significant increase in soil humility due to water can be detected as anomaly and removed from the data [162]. Thus, the ability to explicitly distinguish between anomalies and events is needed.

3.2.2 Device Reliability

Typically, an IoT device is a tiny object that contains four main components: (1) a sensing subsystem to sense and collect information from the environment; (2) a processing subsystem to control all device operations; (3) a communication subsystem to transmit the collected data; (4) a power source to supply the energy needed for all device operations. As we see, the damages in the power source may interrupt the device operations and directly impact the whole IoT system. Unfortunately, the power source is a battery with limited energy capacity. Recharging or replacing such battery is very costly or even impossible, because the devices may be deployed in constrained environments (e.g., under the sewer networks or in the deep forest). In the IoT context, the devices must have a sufficient lifetime to fulfill the application requirements. Therefore, an energy-efficient mechanism for IoT-device has been received special attention from the research community. There are many proposed approaches, but most of them are inherited from Wireless Sensor Network context. Due to the scope of our thesis, we only review data-driven techniques designed to reduce the number of sampled and transmitted data by the device.

3.2.2.1 Related Work

In the context of Internet of Things, the sensing subsystem may consume more energy than the rest of device elements. This is caused by various factors [163]:

- *Power hungry transducers:* Many types of sensors use high power resources to perform sensing tasks, such as multimedia sensors or chemical sensors.
- *Power hungry Analog/Digital converter:* Some sensors need sensing data from analog to digital format.
- *Long acquisition time:* Some sensors may even require seconds to performing sensing tasks.

Reducing the energy for communications by selecting low-power communication technology may be not enough. The energy-efficient methods need also decrease the number of data acquisitions (collecting information via sensing subsystem). Thereby, we also reduce the number of communication as well.

The number of sampling could be reduced by exploiting the correlation in the collected data. The authors in [164] use the temporal analysis to build an adaptive sampling for a snow monitoring scenario. They propose an algorithm based on a modified *CUSUM* [165] that adaptively estimates the current frequency from the trend of historical data. The limitation of such approach is too heavy for constrained-resource devices. A similar algorithm is also proposed in [83], which uses a Kalman filter to calculate the sampling rates. *Spatial correlation* is used in [166] to propose a scheme named "backcasting". The main idea is that the collecting frequencies of data regions having high variation are higher than the others. The same idea is exploited in [167]. Another approach named "hierarchical sampling technique" is applied to IoT devices included various sensor types. This method uses the simple sensors with low-energy consumption in normal situations. When abnormal situations are detected, the complex sensors using high-energy is activated to deal with such situation. This kind of techniques is also called "triggered sampling". In [168], such technique is used to detect a fire emergency scenario. The environment is monitored by low-cost sensors, such as temperature or C02. When a given area has abnormal states (e.g., the increase in temperature or CO2), high-resolution sensors residing in the area are activated. The similar solutions proposed in [169, 170] to detect the state of structures.

In other way, the data prediction is applied to reduce the number of samples by using machine learning models (e.g., linear model, naive bayes). The general idea is that instead of using the sensing subsystem to collect the data, we use a well-trained model to predict such data. The data acquisition is only triggered at low frequency to ensure the model is well-updated. The solution in [171] uses a probability density function as a base model to forecast the next values. The authors of [172] following the same way, but they use a Kalman filter as the prediction model. In [173], a *Dynamic Probabilistic Model* (DPM) is used to implement a probabilistic view of available sampled data. For the time series data, the typical models, such as *Moving Average* (MA), *Auto-Regressive* (AR), and an

Auto-Regressive Moving Average (ARMA) are used as prediction models. PAQ [174] uses an AR model aiming to reduce the computation by the devices. The authors in [175] enrich the AR model to deal with inconsistent data and outliers.

3.2.2.2 Open Challenges

- *Multiple dimensions:* Adaptive sampling is a promised technique to achieve high energy-efficient. However, most of the proposed solutions only optimize the frequency based on a single characterization like in time or space. Thus, we need a comprehensive solution that could combine both time and space to exploit multiple information at the same time.

- *Complexity:* To effectively reduce power consumption, the reviewed approaches process a large amount of data. For example, the data prediction approaches require data and device resources to train the prediction model. The trigger sampling approaches need complex methods and historical data to identify the abnormal situations. However, storing and processing these data on the device side is ineffective for constrained-resource devices. They have to distribute such computation on a device network or cloud instead of a single device. This could emerge the issues related to communication (e.g., How to minimize the cost for distributing data). Therefore, we need a method that is light-weigh and easily deploys on constrained devices while ensuring the high energy-efficiency.

- *Flexibility:* Most of the reviewed algorithm are highly designed for specific purposes. For example, triggered sampling approach more targets in data accuracy than power-saving while prediction approach is contrary. There is no generic algorithm that could handle user's interests related to the power-saving degree. Depending on the user-cases, users could define either data accuracy or power-saving be focused.

Part II

Interoperation in IoT

4
An Industrial IoT Framework to Interoperate IoT Device Connections using Connectors

Industrial Internet of Thing (IIoT) promises many positive impacts on manufacturing, process transformation, and digital value acceleration. However, its benefits are limited to connected devices and open data sources to enrich the measurements and analysis because of the heterogeneity in IoT devices and networks. Several methods have been proposed but large-scale (large volume, device types) device integration is still an open challenge (discussed in section 3.1.2). In this chapter, we propose an innovative IIoT framework that could facilitate creating a connector used in industrial scenarios. Thereby, our solution significantly reduces the efforts in establishing connections from heterogeneous IoT Things to cloud-based IoT platforms. This capability is essential to adapt to the lack of global standards and the rapid changes of the IoT things. In the evaluation section, the scalability and flexibility of our platform are practically demonstrated by satisfactory performance, lightweight software implementation, and low memory consuming.

4.1 Introduction

Over the next decade, the Internet of Things (IoT) will revolutionize manufacturing, energy, agriculture, transportation, and other industrial segments. Its benefits are estimated in the range of \$2.7 - \$6.2 trillion by 2025 [176]. Although Industrial IoT offers infinite potentials and opportunities for current industries, the IIoT interoperability remains a significant challenge due to lack of uniform standards. Each IoT object like sensors and actuators provides different software interfaces and configurations for exchanging data and commands with cloud-based platforms. Furthermore, these interfaces are non-standardized and rapidly changing. For example, in the *Low Power Wide Area Network* (LPWAN) scenario [38], to retrieve the sensed data from IoT devices using LPWAN connections, we have to configure an HTTP callback following a particular format provided by network providers. Because of the missing of global standards, these providers offer their own configuration formats and change them frequently. As a result, there is a giant gap in syntactical interoperability and stability.

In the IoT, data is the vital asset which plays a tremendous role to produce appropriate decisions. More collected data may produce better decisions [177]. Thus, enriching IoT data from external data sources is a promised approach in both academic and industry. However, integrating with *open data services* (ODS) is challenged by the heterogeneity of ODSs. Most of the ODSs use client-server models based on customized RESTful web services which have diverse HTTP configurations (such as path, method, header), data formats (JSON, XML) and data syntaxes. For example: “Accuweather”, a weather data sharing service, configures a API key in the URL namely “apikey”. In contrast, “Openweather” configures this key in the HTTP header under a different name. Therefore, a novel mechanism to deal with heterogeneous syntax of ODSs should be considered.

One of the emerging technologies to achieve interoperability in IoT is IoT middleware. It is a software system implemented as a middle layer between device and application layers. The IoT middleware provides a set of programming abstraction to cover the low-level communication between IoT devices and end-user applications [178]. Global Sensor Network (GSN), Xively, Paraimpu, ThingWorx are some of typical middleware solutions. These systems aim to achieve seamless integration between the heterogeneous IoT Things and applications using different approaches. For example, GSN uses a concept, namely *wrapper*, to handle connections from sensors to the Internet. Hydra, ThingWorx offer a *Device Development Kit* (DDK) to develop applications installed on the IoT Things. However, these approaches require programming skill and many efforts to establish the connectivity to new IoT objects, which do not conform with these middleware in term of data formats or connecting methods. For instance, to establish the connection for novel sensors in GSN, we must build the *wrapper* and a corresponding configuration file for every sensor using provided libraries written by *C programming language* [179].

In this chapter, we present a novel Industrial IoT framework that supports establishing and configuring the heterogeneous connectivity via *connectors.* Typically, the connector is a specific code segment used to wrap heterogeneous connectivity objects in a simple RESTful web service. Thereby, the end-users easily connect to heterogeneous devices via provided web services regardless of the complexity behind. This capability is essential to adapt to the lack of global standards and the rapid changes of IoT things. Our framework provides management APIs to perform full “create, read, update, delete” (CRUD) operations on the connectors. In addition, our proposal may assist end-users in quickly retrieving data from various open data sources based on given connector templates. Notably, our proposed framework is suitable for the LPWAN scenario which is lack of uniform networking standards between service providers [180]. Moreover, collected data from LPWAN devices is very restricted due to the low data-rate connection [181]. Thus, enriching data from open data sources is necessary to maximize benefits from data analysis and context-awareness. The scalability and flexibility of our platform are demonstrated by satisfactory performance, lightweight software implementation, and low memory consuming in practical deployments.

The main our contributions are:

- Identifying current middleware limitations related to integrating with heterogeneous IoT Things in large-scale deployment (huge device volume and various device types).
- Proposing a novel and lightweight framework to accelerate the connecting process for heterogeneous IoT Things.
- Leveraging the proposed framework to speed up the data acquisition from open data sources.
- Evaluating the scalability and flexibility of the proposed framework in real use-cases.

4.2 Related Work

In this section, we review the connectivity mechanism for heterogeneous IoT devices in existing IoT frameworks. We also identify their limitations and our motivations to deliver a novel IoT framework.

FIWARE is cloud-based middleware platform that provides an infrastructure to reduce the cost of creation and delivery IoT services by sharing and re-using *Generic Enablers* (GE) [182]. All API and GE specifications are public and royalty-free for all developers. These documents have necessary information to launch an IoT product that can interoperate with others developed by using the same GE in FIWARE community. The other innovative aspect of FIWARE is that all IoT things are covered behind *OMA Next Generation Service Interface* (NGSI) entities. Therefore, developers only learn and work with the NGSI API used in FIWARE regardless the complexity of IoT technologies and deployment. To handle messages from the IoT devices and gateway, FIWARE provides an element, namely *IoT Agent*, is used to receive and translate the messages to a uniform format. Currently, FIWARE IoT Agent supports HTTP and MQTT protocols [183]. However, creating a new IoT Agent involves hard efforts to follow the FIWARE framework which requires development skills. To integrate with data sharing service, FIWARE proposes *Cygnus*, a connector between *Orion Context Brokers* to FIWARE storages like CKAN, HADOOP, and DynamoDB. Cygnus is based on Apache Flume which supports collecting data via persistence agents. Cygnus only supports some specific HTTP Flume agents. That means FIWARE only integrates with few supported data sources.

Global Sensor Network (GSN) is a platform aiming to integrate with various sensor types. GSN facilitates the connecting process from heterogeneous sensors to IoT applications by using wrappers and XML files [184]. In more detail, these XML files define basic configurations of a sensor such as the data type, parameters, and the corresponding wrapper. The wrapper acts as a sensor driver to establish the connection from sensors to GSN. These wrappers and XML files must be created for every sensor in GSN. However, creating the wrapper are very complicated and requires high programming skills [185]. Another critical drawback of GSN is that all sensor data is stored in an SQL database. Consequently, the GSN performance and scalability are limited.

Hydra is a service-oriented middleware for physical devices, also known as *Link Smart* [186]. This middleware is developed based on *Service Oriented Architecture* that uses web services for seamless integration heterogeneous physical devices into IoT applications regardless of connectivity technologies. Hydra provides a dedicated access control mechanism to ensure the authorization and privacy for all IoT services and devices. The Hydra IoT devices are described by using semantic technologies. Hence, these devices are discovered automatically in the Hydra Network using peer-to-peer network technology. A set of development resources (including the *Software Development Kit* (SDK) and the *Device Development Kit* (DDK)) is also proposed to create applications on both device and end-user sides. However, both SDK and DDK are complicated to non-tech-savvy users. Therefore, this middleware is unsuitable to rapidly create and deploy the IoT devices, services, and applications. Moreover, the Hydra device applications have to interact with middleware using web services. In this way, these applications are extremely heavy to operate on constraint IoT devices which are limited memory and computation capabilities.

Kaa is open-source IoT middleware that supports the users to build complete end-to-end IoT solutions by providing various IoT services, such as data management, data connection, and configuration management [187]. Kaa middleware uses data and configuration schemes to configure IoT devices in term of data structures and device configurations. These schemes are created and managed by *Common Type Library* (CTL). In addition, Kaa provides an SDK to develop the embedded applications for IoT devices in several programming languages like C, C++, Java and Objective C. However, the endpoint SDK only supports few specific IoT devices and the interoperation with other data sources is unmentioned.

Xively is a cloud-based platform using a central message bus to route collected data from IoT devices to other platform components. This platform provides many development tools and resources supported developers to connect and obtain data from their sensors. The IoT sensors can connect to Xively via MQTT, HTTP, and Web Socket protocol [188]. Adding a new sensor supported by Xively is uncomplicated, but the provided API is hard to use, especially for the unsupported sensor. Furthermore, there is no function to integrate with open data sources via web services.

ThingWorx ThingWorx platform offers the IoT applications that are used to monitor, manage, and control connected devices through model-driven development. All sensors, applications, and services are treated as data sources and inter-connected via *virtual buses* [189]. The platform supports several connection protocols including CoAP, MQTT, REST/HTTP, and Web Socket. It also supports integrating with other sharing data sources via web services including open weather services, social data providers. However, ThingWorx only supports few web services and integration with new ones . The other limitation of ThingWorx is that ThingWorx devices only allow connecting to ThingWorx's Cloud by using the applications to be implemented by ThingWorx SDK [190].

Most of the reviewed platforms support very limited sensor types. Moreover, they require specific applications installed on the device side. Adding a new sensor type to these platforms is complex and requires advanced programming skills. There is no mechanism to deal with the rapid changes of IoT devices regarding the software interfaces, connectivity protocols, and data formats. On the other hand, reviewed frameworks do not support the end-users to establish connections and collect data from open data sources via HTTP, MQTT, CoAP, and WS protocols. Our framework has been designed and developed to overcome these limitations.

4.3 IoT Framework for Connectors

This section presents our middleware architecture, connector management mechanism, and connector generation process. At the end, we describe different deployment scenarios to emphasize the high compatibility of our proposal with different IoT system components (e.g., mobiles, gateways, and cloud-based platforms).

4.3.1 System Architecture Overview

Our framework simplifies the creation and management process of heterogeneous connectivity by wrapping the complex establishing and connecting functionalities in the RESTful web services that are efficiently handled by end-users. Figure 4.1 depicts our framework architecture composed of three different layers that are described below.

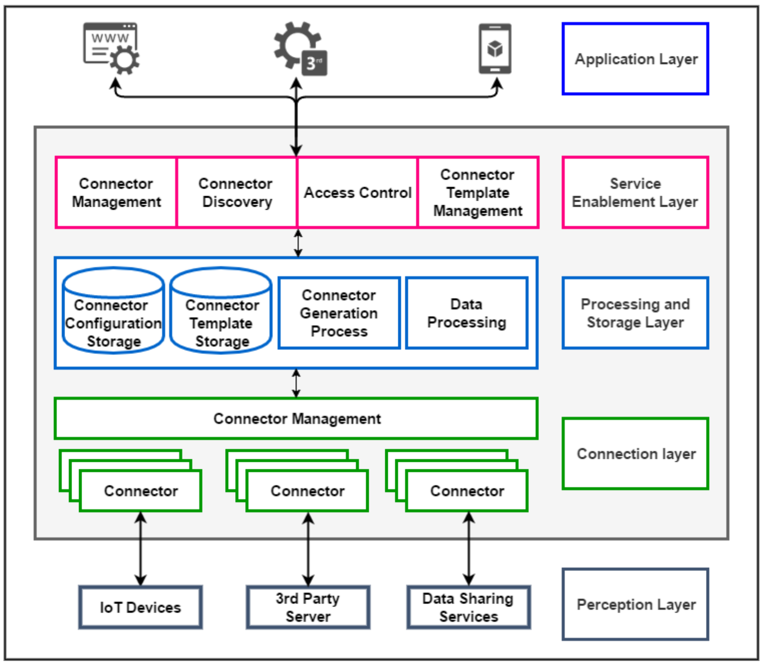


Figure 4.1 – The architecture overview of our framework.

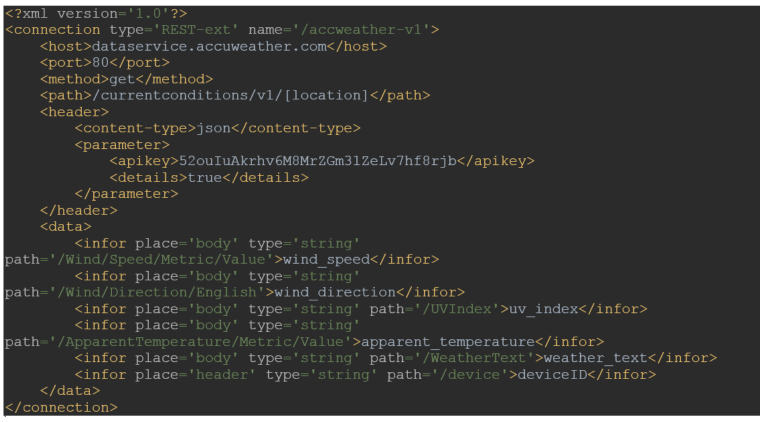
- **Service Enablement Layer:** This layer consists of several web services used to directly interact with the framework. There are four primary services including connector management, connector discovery, access control and connector template management. These services supports: (i) discovering connector, (ii) CRUD operations on connector and connector template, (iii) activating, de-activating connector and (iv) access control based on session token.

- **Processing and Storage Layer:** This layer contains databases and primary functions to generate the connectors and pre-process data. These databases store the configuration of the connectors, connector templates. The functions manages the *Connector Generation Process* and *Data Processing* Services. The details of these services are described in Section 4.3.4.

- **Connection Layer:** This layer consists of several connectors used to handle the connections from IoT Things. Our framework supports two types of connectors - “Connector In” and “Connector Out”. They are analogous to “proxy-in” and “proxy-out” concepts introduced in [191]. Each type of connector supports HTTP, MQTT, CoAP and WebSocket (WS) connectivity respectively to retrieve data from sensor, actuator, gateway and open data sharing services. This is extending the “collection proxies” concept presented in [192]. Moreover, this layer keeps tracking the connector status via connector management module. The end user can manage this status via provide web services.

4.3.2 Connector Generation Elements

To facilitate the connector generation for end-user, we propose *Connectivity Configuration Template* (CTT) files that contains the major configurations of connectivity like connection properties, and data descriptions. This file is encoded under XML format is simplicity, openness, and extensibility. The CTT's content is coherent and convenient for both human and machine. Moreover, the structure of CTT is roughly equivalent to network packets corresponding to supported connections. Fig. 4.2 illustrates an example of a CTT file used to establish an HTTP connection to *Accuweather*, an open data sharing service about weather information.



```xml
<?xml version='1.0'?>
<connection type='REST-ext' name='/accweather-v1'>
    <host>dataservice.accuweather.com</host>
    <port>80</port>
    <method>get</method>
    <path>/currentconditions/v1/[location]</path>
    <header>
        <content-type>json</content-type>
        <parameter>
            <apikey>52ouIuAkrhv6M8MrZGm31ZeLv7hf8rjb</apikey>
            <details>true</details>
        </parameter>
    </header>
    <data>
        <infor place='body' type='string'
path='/Wind/Speed/Metric/Value'>wind_speed</infor>
        <infor place='body' type='string'
path='/Wind/Direction/English'>wind_direction</infor>
        <infor place='body' type='string' path='/UVIndex'>uv_index</infor>
        <infor place='body' type='string'
path='/ApparentTemperature/Metric/Value'>apparent_temperature</infor>
        <infor place='body' type='string' path='/WeatherText'>weather_text</infor>
        <infor place='header' type='string' path='/device'>deviceID</infor>
    </data>
</connection>
```

Figure 4.2 – An example of Connection Template.

From Figure 4.2, the necessary connection properties are a host address, port number, connection method, and path address. The header element defines the HTTP configurations such as content type, authentication information, and HTTP parameters. The next element describes the HTTP payload. Each data in this payload is defined by a "infor" tag that has three attributes: (i) a "place" attribute describes the HTTP object storing the payload, such as HTTP body or HTTP header, (ii) a "type" attribute describes the type of the data like string or number, (iii) a "path" attribute defines the specific location of data in the payload.

The connector is a piece of code, which is used to open connectivity and perform data acquisition. In addition, it anotates the raw data by using define vocabularies which increase the interoperability. In general, the connector structure consists of three distinct parts with the different responsibilities to (i) open the connection based on the received in the CTT file, (ii) de-capsulate and process the data from the received network packet, (iii) manage the connector status via the web services. The connector functionalities are triggered by incoming network packets or defined interval time to obtain data from open data sharing services. The framework effectively manages the connector by using a unique name. In case multiple devices connect to the same connector, their device identities are used. The position of the device identity is defined in the connector content. For example, in Figure 4.2, the device identity is retrieved via "device" parameter in HTTP header. The connector is created by combining CTT and *Connector Template* (CT). CT is a composed script with the marked position which is filled by the extracted information from CTT to create the connector. Each connector type has specific CT.

4.3.3 The Framework Process

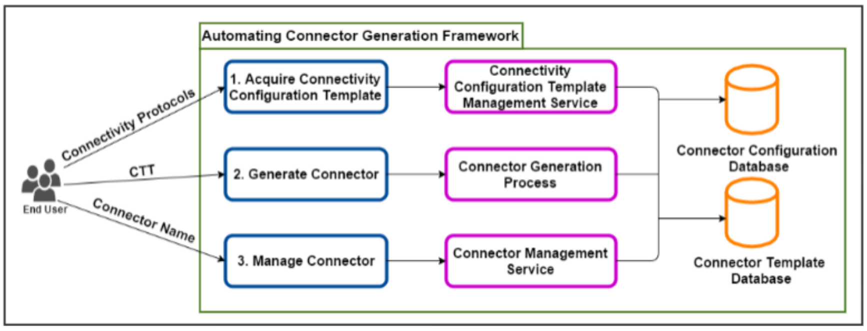

Automating Connector Generation Framework
1. Acquire Connectivity Configuration Template
2. Generate Connector
3. Manage Connector
Connectivity Configuration Template Management Service
Connector Generation Process
Connector Management Service
Connector Configuration Database
Connector Template Database
End User
Connectivity Protocols
CTT
Connector Name

Figure 4.3 – The framework in operation

The primary tasks of our framework are presented in Fig. 4.3. The end-users must follow these steps to create and manage the connectors. Firstly, they send a desired connection protocol (e.g., HTTP, MQTT) to the framework by the web services. Then, the framework responses a CTT file corresponding with the demanded connection. Secondly, the end-users fill the CTT file with connectivity configurations, such as host address, port number before returning to the framework to trigger the connector generation process. Finally, the framework generates the desired connector based on the received CTT file and responses the connector identity to the user.

4.3.4 Connector Generation Process

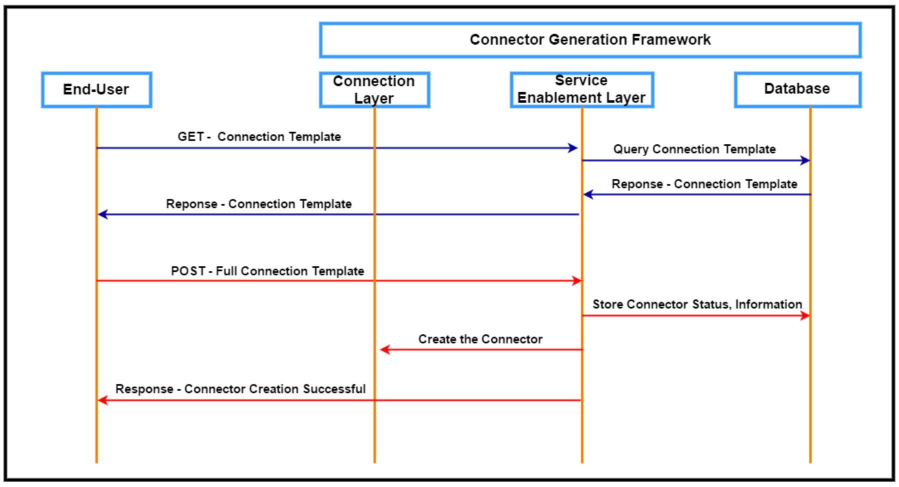

Connector Generation Framework
End-User
Connection Layer
Service Enablement Layer
Database
GET - Connection Template
Query Connection Template
Reponse - Connection Template
Reponse - Connection Template
POST - Full Connection Template
Store Connector Status, Information
Create the Connector
Response - Connector Creation Successful

Figure 4.4 – The operational diagram.

The connector generation is triggered when the framework receives the CTT file via the Service Enablement Layer. To support the non-specialist end-users, the CCT file can be delivered to end-users in advance by using RESTful web services. In the next step, the framework extracts the vital information from received CTT and injects it into proper positions in the CT file. Then, this CT file is executed to generate the connector. In the end, the new connector is stored in the database and registered with connection layer. The general process is illustrated in Fig. 4.4.

4.3.5 Connector Management

To effectively manage the connectors, we propose *Connector Management Component* (CMP) located in the Connection Layer. CMP manages all existing connectors following *Resource Oriented Model* [193]. Each connector is identified via unique name used to discover connector resources via a "Connector discovery" component in the Service Enablement Layer. At first declaration, the framework registers the new connector with CMP using its name. Then, CMP allocates a set of RESTful web services to the registered connector. Our framework supports basic management actions on a connector including CRUD, activation, and deactivation. For instance, after successful creating, a connector is associated with "active" status and ready for operating. The end-user can deactivate the connector by triggering "deactivate" action.

4.3.6 Deployment Scenarios

Our framework is implemented using *Node.js programming language*, a JavaScript runtime built on *Chrome's V8 JavaScript engine* supporting event-driven, non-blocking I/O model [194]. Therefore, it is lightweight enough to deploy in the wide-range *M2M objects*, such as gateways, cloud-based systems, and even smartphones. This capability makes the proposed framework flexible to integrate into various parts of the IoT ecosystem. For a large-scale enterprise using various communication technologies, our framework can be deployed to the gateways to facilitate the connection process for new devices and quickly adapt to the changes of configuration from network providers. In the real scenario, the framework is deployed in a cloud-based system used to simplify connecting process to heterogeneous devices through LPWAN connectivity. In the other perspective, our framework can be implemented as a *data acquisition layer* in other frameworks like SIGHTED [195] or as a proxy layer in a lightweight framework for efficient M2M device management in the *oneM2M Architecture* [196] to accelerate the connection process.

4.4 Evaluation

To evaluate our framework, we measure the execution time of connector generation process including two main operations:

- **Creating connector:** The operation performs the combination of CTT uploading from end-user and CT storing in the database to generate the desired connector.
- **Reloading framework:** After connector generation process, the framework is reload to integrate new connectors into the framework.

The evaluation is performed on-device contains the processor: Intel(R) Core(TM) i5-6200U CPU @ 2.30 GHz, 2401 MHz, 2 Core(s), 4 Logical Processor(s), 4 GB of RAM and the operating system is 64-bit Windows 10. The acquired result is shown in Figure 6. According to that, reloading framework takes around 1.15 s to 2.91 s while creating connector only consumes around 0.113 s to 1.4 s. On average, the total execution time is from 1.265 s to 4.31 s. This performance satisfies the user experiment [197]. When the connector works as a "proxy out", the time of establishing a connection is minor around 0.003 ms comparing with the total time largely depended on the performance of the data sources. Also, most of the open data sources are limited in the performance, and the number of requests could be served per second.

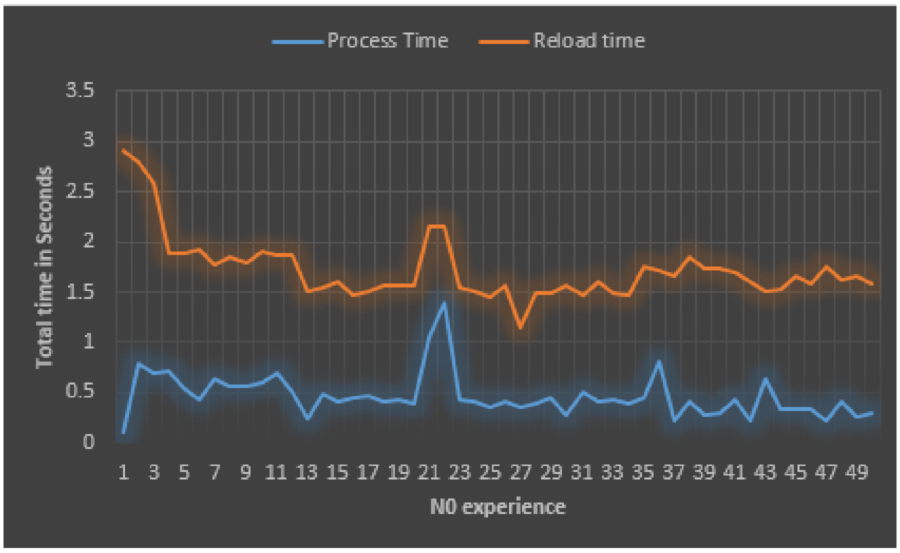


Figure 4.5 – Connector generation performance

The size of connectors for each M2M connectivity is less than 1 KB, and the running memory of our framework is only a couple of megabytes of memory. In this manner, our framework is satisfactory to deploy on the general IoT objects from cloud-based middlewares, M2M gateways and event smartphones with gigabytes of internal memory along with the powerful processor. On the other hand, the size of the CTT only depends on the data section. However, this section is unprocessed or participated in the connector generation process. Therefore, the performance of connector generation process is independent of the size of CTT. Furthermore, the connectors are developed *Node JS Express framework* supporting non-blocking I/O model, and consequently, there is unlimited in the number of devices connecting to a connector. These properties make the framework more scalable and flexible to be deployed in a large-scale scenario.

4.5 Conclusion

In a nutshell, we identify the limitations of existing middleware about integrating heterogeneous connections from IoT Things. Motivating to bridge the gaps, we propose a framework that simplifies the connecting process from heterogeneous IoT Things to cloud-based platform via the connectors. In addition, our platform provides well-supplied management web services based on the resource-oriented model. It allows the end-users to easily discover and perform wide-range management operations on the connector, such as creating, retrieving, updating and deleting as well as activating or de-activating. Another innovative aspect of the connector is to facilitate and speed up the data acquisition process from open data sharing web services. In the evaluation section, the scalability and flexibility of our platform are practically demonstrated by satisfactory performance, lightweight software implementation, and low memory consuming. Regarding future works, we are working on integrating our platform into the oneM2M architecture and implementing an access control mechanism.

5
A Scalable IoT Framework to Maximize Data Knowledge using Virtual Sensor

During recent years, we have witnessed an explosion on the Internet of Thing in term of the quantity and types of physical devices. However, there are many limitations of these devices regarding their computing power, storage, and connection. They affect on-device processing and interoperability of sensed data significantly (discussed in Section 3.1.2). Centralized treatment of IoT data has proven challenging for many use cases demanding response in real-time. This chapter aims at augmenting sensor data processing using the concept of virtual sensors. We present a scalable virtual sensor framework that supports building a *logical data flow* (LDF) by visualizing either physical sensors or custom virtual sensors. The process produces high-level information from the sensed data that can be easily perceived by both machines and humans. A web-based *virtual sensor editor* (VSE) is also implemented on the top of the framework to simplify creating and configuring the LDF. The VSE supports cross-platform and verifying the composed LDF in real-time. Furthermore, we present a taxonomy of supported virtual sensor types along with preliminary performance study.

5.1 Introduction

According to Wikibon report [198], by the end of 2020, 212 billion IoT smart objects are expected to be deployed worldwide. Despite the rapid growth, there are critical challenges to gain high-level information from sensed data. Even the exponential growth of IoT smart objects about processing power and functionality, many situations cannot be handled at the device level, such as (i) a query to check average humidity in a region, (ii) Predicting missing data based on the historical data. To handle these situations, the virtualization of the physical sensor on cloud environment, namely *Virtual Sensor* (VS), is considered as a practical approach. VS is a logical reflection of one or a set of physical sensors on the cloud-based platform and can handle complex tasks which cannot be performed on physical sensors [199]. The major VS benefit is to facilitate and enrich the functionalities of physical sensors at the software level to adapt to different purposes and scenarios [200]. For example, in the tank monitoring use-case, a ultrasonic sensor is plugged in the tank cap to monitor the current liquid level. This sensor only measure the distance from liquid surface to the top of the tank. To measure the liquid volume in a tank, the end-users need to develop a dedicate application and setup the tank information (tank size, liquid types) for this sensor. These works are very complicated and costly. Using virtual sensor, the liquid volume is calculated on cloud via web-based applications [201].

In this chapter, we present a *scalable virtual sensor framework* (sVSF) that simplifies creating and configuring VSs with the programmable operators (such as rules, formulas, and functions). These VSs are linked together to create a hierarchical topology, namely *logical data flow*, to produce high-level information from collected data. In addition, this information is formed under JSON-LD format which is used to generate interpretable data across different IoT platforms [28]. Thereby, the interoperability of our approach is significantly increase. On the top of the framework, a Virtual Sensor Editor is also implemented to facilitate building and configuring the LDF by offering drag-drop actions on a HTML5 web interface. To achieve scalability and performance, sVSF is implemented based on clustering architecture along with various strategies, such as executing LDF following asynchronous model and using No-SQL database.

Our framework supports various virtual sensor types at *Infrastructures as a Service* (IaaS) and *Platform as a Service* (PaaS), such as singular, accumulator, aggregator, selector, qualifier, context-qualifier, and simple predictor [202]. In the sVSF, the outcomes of VSs are stored in the database. Thus, each VS has historical data for further analysis. Furthermore, the proposed framework is remarkably suitable for Low Power Wide Area Networking scenario where bandwidth (12 bytes in SIGFOX and up to 250 bytes in LORA), and data rate (typically 10 kilobits per seconds) are very limited [203]. These restrictions reduce the quantity of collected data that directly affects on the accuracy and trustworthiness of further data analysis. To ensure the scalability and performance, several technologies and strategies are exploited. For example, the core framework is implemented based on clustering architecture using *Nodejs* language. Our work has four highlighted contributions:

- Identifying the limitations of current virtual sensor frameworks in term of VS functionality and usability.
- Reviewing the concept of the virtual sensor and its taxonomy.
- Presenting a scalable virtual sensor framework aiming to increase information quality and interoperability from sensed data.
- Improving the scalability and performance of the proposed framework by implementing the clustering model along with various strategies.
- A cross-platform development tool for the virtual sensor is offered to speed up designing logical data flow.

5.2 Related Work

In this section, we review the virtual sensor concept and existing virtual sensor frameworks in the IoT. Their limitations are also identified. At the end of the section, we present our motivations to bridge the gaps by delivering a scalable IoT virtual sensor framework.

As a definition in [200], a virtual sensor is a reflection of a physical sensor that obtains and represents collected data on the cloud. Following [204], a virtual sensor is an emulation of a physical sensor which collects its data from underlying physical sensors. There are many ways to define and categorize the virtual sensor, in [200], the authors also separate the virtual sensor into two types: (i) Task level: represents the physical sensor as a virtual object that could be processed, calculated. (ii) Node level: represents a subset of physical sensors as a virtual topology. In [204], the virtual sensor is classified into four typical types: (i) One-to-Many: One physical sensor is represented by many virtual sensors. (ii) Many-to-One: One and more physical sensors are presented by one virtual sensor. (iii) Many-to-Many: This is the combination of two described types. (iv) Derived: One virtual sensor represents different physical sensor types. While in other types, the virtual sensor represents the same physical device type. At the IaaS level, [205] categorizes virtual sensors based on their offering services. Therefore, in the IoT, the definition and taxonomy of the virtual sensor are chaos and heterogeneous.

The authors of [206] propose a web-based virtual sensor editor tool to facilitate designing the virtual sensor process. This tool visually aggregates either the physical sensors or customized virtual sensors. It supports calculating and visualizing sensor values on graphic charts in real-time. VSs are created by aggregating physical sensors. The graphic interface supports native HTML5 drag-drop actions and real-time virtual sensor evaluation. As a result of HTM5 characteristics and call-by-need strategy, this tool enables cross-platform and scalability. Similarly, the authors in [207] present a web-based interactive framework to visualize and authorize sensors and actuators for indoor scenarios. Each IoT Thing serves as a node, visualized within a 3D indoor scene. Hence, the end-users can monitor, link and program the sensors and actuators, respectively. This framework bases on a event handling model which treats incoming data as an event. To handle complex events, which must be processed on multiple sensors, the author proposes a hierarchical graph for visual summarizing sensors, actuators, and their relations.

In the IoT context, physical sensors are distributed and affected by many adverse factors including deployment scale [21], sensor constrained resources [22], and intermittent loss of connection [23]. As a result, sensed data need to be processed, filtered and transformed to ensure the precision of collected information. Many middleware platforms are designed to process the IoT data on either multi-sensors or multi-stream [208] [209]. These works aim to improve the information quality of data acquired from heterogeneous data sources. In the same scope, the authors of [210] present a virtual sensor environment that handles sensing data processing in real-time. This approach uses *Complex Event Processing* (CEP) as a virtual sensor engine. Their primary contributions are to capture the benefits from the CEP to define the custom analytic algorithms along with data analysis blocks on the incoming data. The authors in [200] address challenges about implementing virtual sensor at *Software as a service* (SaaS) and *Platform as a service* (PaaS) levels. They propose a sensor-cloud architecture including four separate modules to handle various tasks, such as sensing, processing, storing, and communicating. Each module has an API to build applications and share sensed data to either the IoT users or services.

The limitations of state-of-the-art are described below.

- In [206], we notice limited functionalities for the virtual sensors. These functions only perform on incoming data. In other words, the virtual sensors can not link together. There is no discussion regarding virtual sensor types and which types are supported by their tool.

- The authors in [207] have not shown that how to configure the algorithms for CEP. Moreover, they do not offer a graphic interface to facilitate the configuration process for the data analysis blocks. Similarly, the works of [208, 209] more focus on services and implementation than simplifying configuration process at the user level.

- The work of [210] is used for the indoor scenario. There is no mention on the mechanism to create and configure custom virtual sensors and actuators.

From all points above, there are no comprehensive solution proposing a robust virtual sensor framework along with a friendly web-based interface. In addition, leveraging virtual sensors to present historical data is not supported. Our framework is designed and implemented to mitigate these limitations.

5.3 Virtual Sensor Framework

In this section, we present the definition and taxonomy of the virtual sensor as well as our virtual sensor framework architecture. Such framework is designed based on layered architecture and operates over clustering and asynchronous model to maximize scalability and performance. At the end of the section, we describe the work-flow of the proposed framework in specific deployment scenarios to emphasize its benefit.

5.3.1 Virtual Sensor Definition

We define the virtual sensors as virtual objects, which have operators to perform specific functionalities. In our proposal, we support three operator types including rules, formulas, and functions. We propose a novel taxonomy of the virtual sensor based on its operator. For instance, a virtual sensor is defined as an "Accumulator" type if its operator is an accumulation function. The list of supported virtual sensor types is described below:

- ***Singular***: This sensor type is a one-to-one mapping between the physical sensor and its reflected interface on cloud. Through this virtual interface, the end-user obtain sensed data from physical sensors.

- ***Accumulator***: A virtual sensor performs accumulation functions on its sensing data within a particular duration. For example, a physical sensor measuring rainfall uses the counter value to identify rainwater volume. An accumulator VS is useful to present the rainwater volume within 24 hours by accumulating on these counter values.

- ***Selector***: A virtual sensor represents sensing data from one or many physical sensors based on defined criteria. For instance, a selector virtual sensor represents all temperature data that is higher than 10 degrees.

- ***Aggregator***: A virtual sensor performs basic statistics (such as averaging, maximum, and minimum) on collected data from different physical sensors. For example, in the case of humidity sensors deployed in various regions, an aggregator sensor is used to calculate the average humility of a specific area.

- ***Qualifier***: The same with the singular type but this virtual sensor type only is activated by a qualifier function, which is an IF ELSE statement. For example, one qualifier virtual sensor monitoring temperature can generate an alert when sensing value higher a defined threshold.

- ***Context-qualifier***: The same with qualifier type but the qualifier function performs on a bundle of sensors.

- ***Predictor***: This virtual sensor predicts next sensing values based on analyzing historical data. Such virtual sensor is necessary in case of occurring error on the physical sensors.

- ***Compute***: A virtual sensor has a complex function analyzing the sensing data from a set of the sensors. For example, a compute virtual sensor offers the car statuses based on observed data (such as engine temperature, oil level, and gas level) from car's sensors.

On the other hand, we present a novel component, namely "logical data flow", which represents a chain of virtual sensors to perform a specific task. For example, a logical data flow is used to identify the remains of liquid in a tank from ultrasonic sensor data. This LDF is a chain of one singular virtual sensor, one selector virtual sensor, and one aggregator virtual sensor.

5.3.2 Architecture Overview

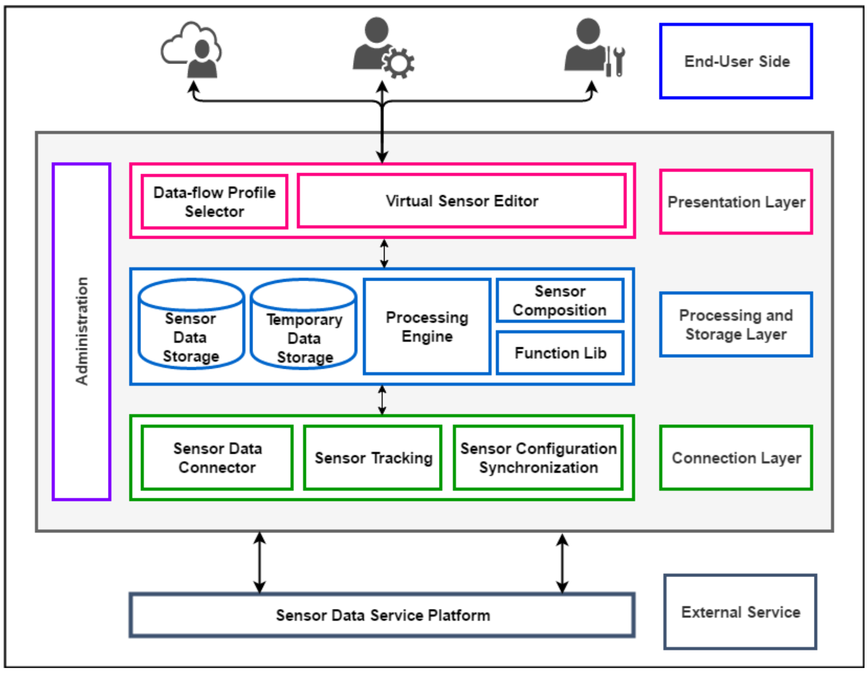


Figure 5.1 – The framework architecture overview.

Our goal is to produce the high-level information from sensing data. To achieve this goal, the framework simplifies creating and configuring the logical data flow by offering an interactive virtual sensor editor. Fig. 5.1 depicts the framework architecture composing of three horizontal layers described below.

- ***Connection Layer***: This layer handles the connections between sVSF and *Sensor Data Service Platform* (SDSP) which aggregates and pre-processes collected data from physical sensors. There are three components: (i) *Sensor Data Connector*: This component is used to interact with SDSP through RESTful web services and MQTT. (ii) *Sensor Configuration Synchronization*: This component synchronizes sensor configuration between sVSF and SDSP. In addition, the configuration of a singular VS will be applied to corresponding physical sensor managed by SDSP through calling a RESTful web service. (iii) *Sensor Tracking*: This component is used to track the new sensors recently registered to SDSP. These sensor profiles are transferred and stored at sVSF.

- ***Processing Layer***: The main components of this layer is databases and a primary engine to execute the logical data flow. There are two databases: (i) *Sensor Data Storage* database permanently stores sensor information and LDF. (ii) *Temporary Data Storage* database stores temporary outcomes of virtual sensor as well as intermediate results of LDF. Such data will be removed after a short duration configured by administrators. Apart from these components, the processing layer has a "Sensor Composition" (CP) component to retain a record of configuration and relationship among virtual sensors at the presentation layer. When a new virtual sensor is created by dragging and dropping onto VSE, such changes are caught and stored by CP. In addition, the CP ensures the logical data flows executed following the asynchronous model. The final component is a *user-defined function library* storing the custom functions declared by end-users. These functions are executed directly from CP.

- ***Presentation layer***: This layer is used to render an interactive HTML5 web interface, namely *virtual sensor editor*. The editor supports the end-user to create a logical data flow. The virtual sensors are visualized as linkable boxes that can be configured either functionality or appearance via a setting panel. These boxes can be linked together to create a logical data flow. All such operations are handled by "Sensor Composition" component in the under layer. Moreover, the presentation layer has a "Data-flow Profile Selector" component used to select and reuse the existing virtual sensor templates and LDFs.

- ***Administration layer***: This layer authorizes and supervises the user access right on VSs and LDFs. Each user type is only able to perform specific actions based on its roles. For instance, a standard user cannot delete a LDF. The administration layer is also used to manage general settings of the framework, such as the time life of temporary data, sensor configuration synchronization interval.

5.3.3 Virtual Sensor Framework Workflow

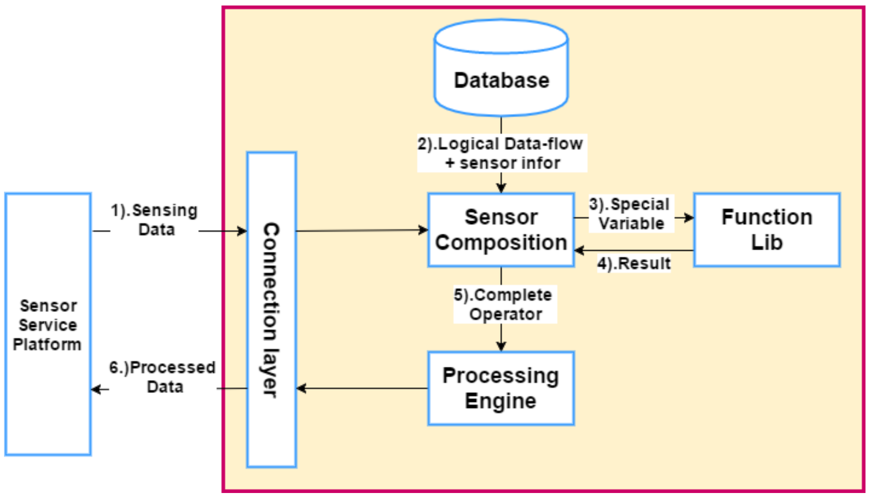


Figure 5.2 – The framework operation diagram.

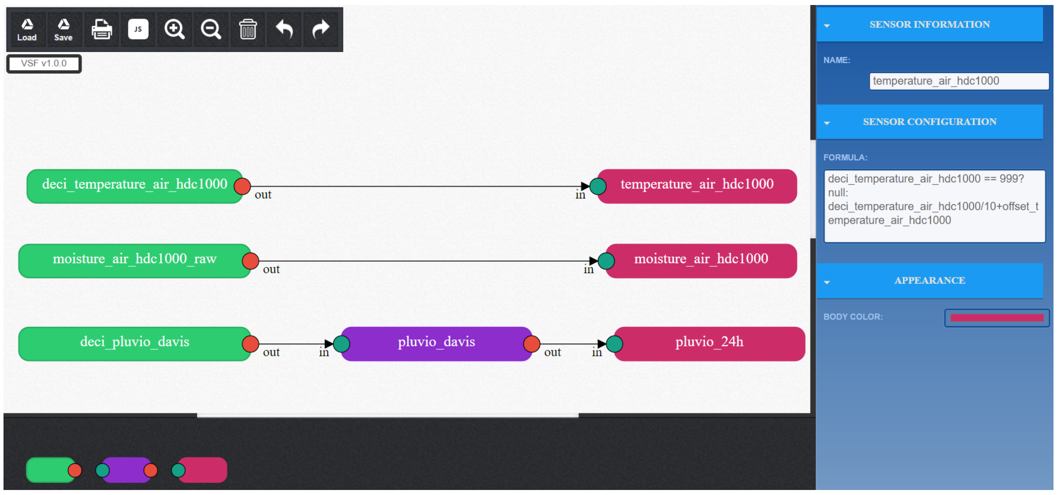
The overall blueprint of our sVSF workflow is shown in Fig. 5.2. Initially, physical sensors register their resource descriptions under *CoRE Link Format* [211] with SDSP. Once successfully registering, all sensing data from registered sensors will be forwarded to sVSF.

These sensors and their resources are discovered through simple search queries. For instance, a query searching all temperature sensors is “GET /rd-lookup/ep?rt=temperature”.

In sVSF, the incoming data is handled by the connection layer. Then, it is conveyed to the Sensor Composition element which processes the VS operators and relationships in the logical data flow. This component also takes responsibility to convert VS operators to the proper mathematical operations and ensures it is executed in the correct order in the processing engine. The conversions of VS operators into mathematical operations pass through two phases: In the first phase, the logical data flow and sensor information are loaded into CP. In the second phase, the particular variables of the operators are extracted and calculated by executing the corresponding functions in *Function Lib Component* (FL). These variables represent particular operations. For example, as shown in Fig. 5.4, $device.tank_level_change.data[24] represents all sensing data of “tank_level_change” sensor within 24 hours. After calculating, the values of special variables are added into the virtual sensor operators before storing in the temporary data storage database to reuse in another stages. The lifetime of this temporary data is set up by administrators. In case the virtual sensor has the input from another virtual sensors. This input value is also considered as a special variable and directly access via the sensor name. As example shown in Fig. 5.4, a sensor, namely “tank_volume”, use the output value of “tank_level” sensor via declaring a special variable named “tank_level.” Furthermore, to maximize performance, Sensor Composition organizes a working schedule based on the asynchronous model. The non-relational virtual sensors, unlinked together, are arranged into the same thread and executed in parallel. For example, in Fig. 5.3, all green virtual sensors are performed in parallel. In the next step, the virtual sensor operator is executed in a processing engine developed by a JavaScript library, namely *MathJS* . Finally, the output of processing engine is stored in the database before responding to SDSP.

5.4 Virtual Sensor Editor

One of a key element of sVSF is a virtual sensor editor operating on web browsers. Thereby, it is compatible with any device or platform. We develop this editor based on WYSIWYG (what you see is what you get) architecture. It allows the user to directly build a LDF by creating, configuring and connecting VSs. VSE is implemented using native HTML5 and JointJS library to maximize portability and availability across various end-user platforms. There are four highlighted attributes of this editor: (i) Drag-drop interface: The end-users can create and link virtual sensors by drag-drop actions. (ii) Real-time evaluation: The logical data flow could be evaluated in real-time. (iii) Reusability: The VS configuration and LDF are stored and shared between the end-users.



NAME: temperature_air_hdc1000

FORMULA:

```
deci_temperature_air_hdc1000 == 999?
null:
deci_temperature_air_hdc1000/10+offset_t
emperature_air_hdc1000
```

Figure 5.3 – The Virtual Sensor Editor interface.

As shown in Fig. 5.3, virtual sensors are represented as linkable boxes consisting of input, output ports, and a VS operator. The number of these ports and the operators depend on the virtual sensor type. For example, a singular virtual sensor, which serves as a physical sensor, has one output and no input port. In the configuration panel, a drop box allows the user to configure sensor drivers. By default, we use the green and red color to identify virtual sensor type. However, the end-user can change this attribute in the configuration panel. Each output port can be assigned to one or more input of several boxes. After linking, the later sensor could select the output of former sensors as an input parameter of its operator. The auto-complete feature is also implemented to speed up the selecting process.

Our sVSF offers recursive composition for the virtual sensors. In detail, a virtual sensor can be used as an input to construct other virtual sensors. After logical data flow is completely established, the evaluation feature allows the user to execute this data flow on sample data sets and receive the result immediately. Hence, the users can evaluate or correct the LDF configuration in case of errors. At the final state, the complete logical data flow is saved in the database and reused for next time.

5.5 Evaluation

In this section, we describe the utilization of our framework in a practical use-case. We also discuss how to achieve high performance, scalability in our framework. At the end, an evaluation of our solution is proposed.

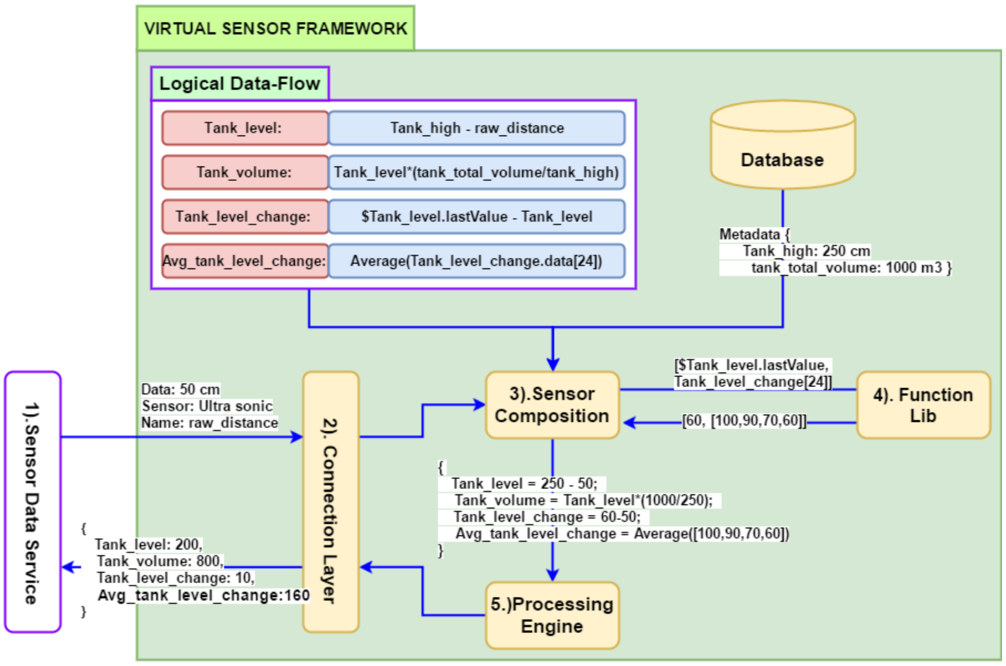

VIRTUAL SENSOR FRAMEWORK
Logical Data-Flow
Tank_level: Tank_high - raw_distance
Tank_volume: Tank_level*(tank_total_volume/tank_high)
Tank_level_change: $Tank_level.lastValue - Tank_level
Avg_tank_level_change: Average(Tank_level_change.data[24])
Database
Metadata { Tank_high: 250 cm tank_total_volume: 1000 m3 }
1).Sensor Data Service
Data: 50 cm
Sensor: Ultra sonic
Name: raw_distance
2). Connection Layer
3).Sensor Composition
[$Tank_level.lastValue, Tank_level_change[24]]
[60, [100,90,70,60]]
4). Function Lib
{ Tank_level = 250 - 50; Tank_volume = Tank_level*(1000/250); Tank_level_change = 60-50; Avg_tank_level_change = Average([100,90,70,60]) }
5.)Processing Engine
{ Tank_level: 200, Tank_volume: 800, Tank_level_change: 10, Avg_tank_level_change:160 }

Figure 5.4 – The generating high-level information process.

Our work is used in an industrial project for monitoring tanks. We aim to manage the chemical volume via the level sensors plugged at the top of tanks. The sensed data represents the distance from the top of a tank to chemical surface. Figure 5.3 illustrates the whole process of generating high-level information such as remaining of the chemical level (Tank_level), remains of chemical volume (Tank_volume), change of chemical level (Tank_level_change), average on this change within 24h (Avg_tank_level_change). Firstly, the raw data and sensor metadata are sent to sVSF. This data is handled by Sensor Composition where corresponding logical data flow and sensor metadata are loaded from the database. In this example, the metadata is the tank information, such as the height of tank (Tank_high) and the total volume of tank (Tank_total_volume). In addition, individual variables such as last value of such sensors (Tank_leve.lastValue) or historical sensing data within 24h (Tank_leve_change.data[24]) are calculated by calling the corresponding functions in Function Lib component. All obtained information (e.g., special variable value, raw data value, and sensor metadata) is injected into the logical data low. Before transferring and executing at the processing engine, the logical data flow is converted to mathematical operations. The final result is responded to SDSP via Connection Layer.

SVSF is developed by integrating MathJS library into NodeJS Express framework that supports the event-driven architecture. Nodejs also leverages the non-blocking I/O model allowed requests being processed asynchronously [212]. In order to enhance the framework scalability, we exploit the clustering architecture. A cluster comprises a set of servers running simultaneously. Each server is called a node. The cluster is elastic to adapt to the unexpected changes in term of the number of the concurrent users by dynamically add or remove nodes to the cluster. There are two types of nodes: master node and worker node. The master node is used to distribute requests among different worker nodes in the cluster. Other strategies are proposed to increase the performance:

- The first strategy is to store the outputs of the virtual sensors in a temporary database.

Such values could be re-used as the input of other VS sensors instead of re-calculating.

- The second strategy is to use an in-memory database to speed up data querying process. Our framework uses a NoSQL database named Apache CouchDB [213]. Comparing with a relational database, CouchDB stores the data in independent documents and its self-contained schema. As a result, it provides massive scalability and powerful full-text search.

- The final strategy is to adopt the asynchronous model to execute logical data flow, meaning that all independent virtual sensor or virtual sensor in a similar stage are executed in parallel.

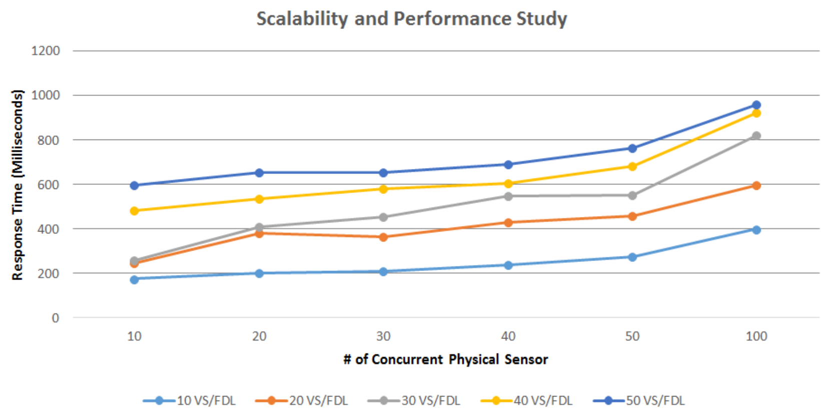


Figure 5.5 – The framework's performance

To evaluate the effectiveness of the proposed strategies, we have to consider two scenarios: (1) Significantly increasing the number of simultaneous physical sensors in SDSP; (2) Increasing the complexity of logical data flow regarding the number of VSs. All evaluations are performed on a computer with following configuration: Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz, 2401 MHz, 2 Core(s), 4 Logical Processor(s), 8GB of RAM and the operating system is 64-bit Windows 10. The clustering model is set up and deployed using *native clustering module* provided by NodeJS. The data rate of the physical sensor is one message per second. For the first scenario, we have experimented with different scales of sensor network, which increase from 100 to 450 concurrent physical sensors. The logical data flow comprises 50 virtual sensors. Fig. 5.5 illustrates the performance changes after adopting our enhancements. As shown in the figure, our enhancement is remarkably effective. Without clustering model and enhancement strategies, the response time is significant increase when expanding the scale of sensor network. After applying clustering model using 4 or 8 clusters, the response time is highly stable under 1 second regardless the size of sensor network. With 450 concurrent physical sensors, the normal response time is over 6 seconds comparing with 875 milliseconds and 650 milliseconds of the model using 4 and 8 clusters respectively.

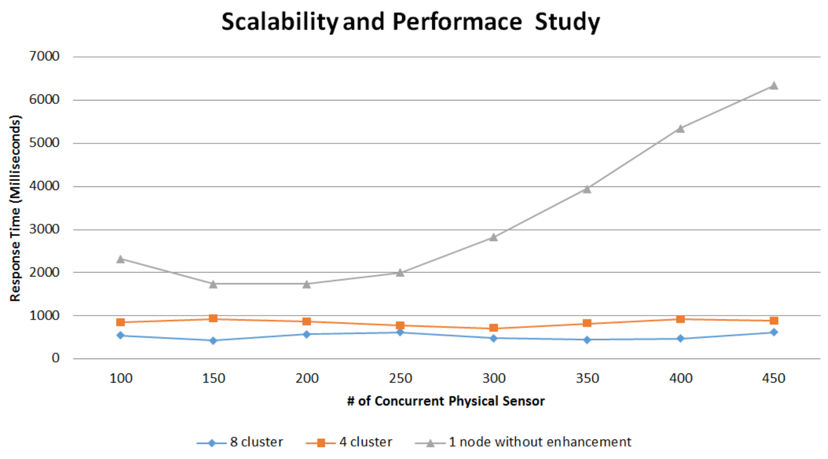


Figure 5.6 – The effect of our enhancement in scalability and performance.

In the second scenario, the simulations are performed with different logical data flows size, which contains from 1 to 100 virtual sensors. Each logical data flow is evaluated by various sensor network scale in SDSP. As shown in Fig. 5.6, when increasing the number of concurrent physical sensor, the response time lightly increases regardless of logical data flow size. In case the scale of the logical data flow is moderate (comprising under 50 virtual sensors), our system serves a data message under 800ms even when 50 concurrent physical sensors are running. In the case of scaling up to 100 concurrent physical sensors, the response time is still under 1 second.

5.6 Conclusion

In this chapter, we review the virtual sensor concept and propose a new virtual sensor taxonomy based on its functionality. The limitations of the existing virtual sensor frameworks are considered in term of virtual sensor functionality and usability. Motivating to bridge the gaps, we proposed a scalable virtual sensor framework producing high-level information from sensed data, by creating a logical data flow over a set of virtual sensors. A web-based virtual sensor editor is also offered to accelerate creating and configuring the logical data flow. In the evaluation section, a serial of strategies to enhance the performance and scalability is discussed and evaluated. For future works, we are currently working on integrating our platform into the oneM2M based framework [214] and FIWARE architecture to ensure the interoperability.

6 WoT-AD: A Descriptive Language for Group of Things

Recently, the *Massive Internet of Things* (Massive IoT) and *Web of Things* (WoT) was remarkable research fields aiming to facilitate the connectivity and accessibility of the IoT Things by leveraging Web standards and technologies. In such context, the end-user is capable of simply creating, mashing-up and presenting the multiple Things to gain high-level information. However, current approaches pay much attention to describe a single Thing. The modeling and building the application for *groups of Things*, namely "Asset", are still limited due to the lack of descriptions and seamless integration mechanisms. Moreover, the traditional *IoT Device Description Language* directly installed on device is highly restricted in Massive IoT scenario because of stringent requirements about power consumption and operation costs. In this chapter, we introduce the *WoT based Asset Description* (WoT-AD), a descriptive language for the Asset aiming to mitigate such limitations. WoT-AD semantically describes a group of Things as a homogeneous object that supports to mash-up, discover, and access their resources, entities, and services. In addition, we provide a lightweight framework fully integrated with WoT-AD to maximize the semantic interoperability. This integration also simplifies the development of mash-up applications to different-skilled users. Finally, we evaluate the performance of our proposal to demonstrate its effectiveness and scalability in real use-case.

6.1 Introduction

With the exponential growth of the number of Things, by the end of 2020, 212 billons IoT smart objects are expected to be deployed worldwide [215]. *Machine-to-Machine* traffic flows will constitute up to 45% of the whole internet traffic in 2022 [216]. As a result, Massive IoT (MIoT) is emerging as a novel technology referring to the massive volume of constrained IoT devices which stringently require excellent coverage, cost-effective, and low-energy consumption [217]. Among several emerging connectivity technologies for MIoT, proprietary LPWAN technologies such as Sigfox and LoRA have been considering the most potential candidates while cellular-based connectives such as 5G or NB-IoT are under developing and testing process [38]. Although these protocols support collecting data from Things through the web protocols, the access and mash-up the Things resources are still limited due to lack of global standards [218]. Hence, IoT Things is typically formed into small and isolated silos for specific applications.

These considerations leverage to Web of Things concept for directly collecting and accessing any Thing's resources as web resources. However, to fulfill Massive IoT requirements, there are some existing drawbacks mainly related to the constraints in power and computing capabilities[219]. For example, the MIoT Things can not run WoT applications, which are costly to constrained devices. In a similar situation, other effective solutions based on pub/sub model suffer from the limits of LPWAN down-link. Most of them require the bi-directional connection to update the Thing's resources whereas LPWAN connectivity only supports a limited number of down-link messages per day [220].

At a higher level, the WoT is expected to break-down the *Silo of Things* by combining the traditional web paradigms with unified standards to present the Things. This integration allows the Things to be published, managed, and directly accessible. To do so, logical interfaces of Things are used to present the Things resources and services using semantic web languages and annotations, such as the Device Description Language (DDL) [221], IoT-DDL [222], WoT-TD [223], CoRE-TD [224]. However, all mentioned approaches target to describe a single Thing that has the limited number of sensors and services. In practice, the Things may be a compound object also called "Asset" including the hierarchical group of Things and services. For example, in the smart building scenario, the Asset could be a floor that has various rooms monitored and controlled by several IoT devices. Each room is considered as an independent Thing with different resources and services tightly linked to these IoT devices. Therefore, we need a descriptive language to describe either single or compound Things effectively. In this way, we enhances the capability of access and managing the IoT Things.

In this chapter, we propose an approach enabling the semantic interoperability for the Asset in Massive IoT scenario. Our method is based on a novel semantic description, namely WoT-AD. We also provide a light-weight WoT framework fully exploited the benefits of WoT-AD concept. Such combination is not only capable of presenting, accessing, and managing the Asset but also speeds up the IoT application development. According to our solution, the Asset composing a group of Things is modeled as a uniform object that supports discovering, querying and visualized its resources through an unified interface. In addition, creating Asset is facilitated to non-tech-savvy users by using a web-based graphic interface. This increases the usability of our proposal and make it suitable to a wide-range of Massive IoT use-cases. Our architecture consists of four primary layers as illustrated in Figure 6.3. The top of architecture is a *composition layer* that directly interacts with users. This layer supports composing the user's desired Asset from the available templates. The next layer named *execution layer* is used to (1) generate Asset API; (2) execute of Asset Model; (3) index the physical devices. The fundamental component of this layer is virtual sensor framework which receives, processes and updates the Asset's resources from collected data. The lowest layer is the connectivity layer used to handle the connections from various IoT devices as well as external data sources (e.g., weather forecast, open IoT stream). The designed architecture is implemented as a framework based on the clustering model to enhance scalability and performance. We experimentally evaluate proposed architecture in a smart space scenario. In summary, our contribution is highlighted following:

1. Identifying the limitations of existing Things descriptions and WoT frameworks about describing and managing the Asset.

2. Proposing a novel semantic description for the Asset to mitigate the existing limitations.

3. Presenting a lightweight framework enabling the semantic interoperability for Massive IoT scenario.

4. Implementing a cross-platform development tool on the top of the framework to speed up the Asset development process for non-tech-savvy users.

6.2 Background Analysis

In this section, we briefly review the Things description and existing Web of Things frameworks. Their limitations are also identified. At the end, we present the motivations to bridge the gaps by delivering a lightweight WoT framework along with a novel semantic description language for group of Things.

6.2.1 Things Description

To enable the IoT device integration in smart space scenario, the Mobile and Pervasive Computing Lab at University of Florida presents a *Device Description Language* (DDL) based on the Service Oriented Architecture model. The IoT device in DDL is described as an entity including properties, internal mechanisms, and interfaces [221]. Initially, the DDL is integrated within the *Atlas sensor platform.* Then, it is used to develop the *Cloud-Edge-Beneath* (CEB) architecture [225] to handle a large volume of sensors and devices in smart city context. This CEB is based on Atlas architecture supporting end-user to directly access the IoT devices or sensors through cloud interfaces [226, 227]. The Atlas framework and DDL are implemented at Edge devices as an intermediate layer between cloud and "Beneath" layers. The author uses the *Open Service Gateway Initiative* to connect to Atlas sensor platform and discovery services.

The *World Wide Web Consortium* provides a WoT framework aiming to abstract the Things through a set of web services. To describe the Things, W3C proposes the *W3C Things Description* (WoT-TD) consisting of (1) semantic description to present the general information of Things; (2) an interaction model with WoT properties, Action, and Event to present the Thing's resources and services; (3) a semantic schema to express the data model [228]. The WoT-TD is used in [229] to describe in detail about how to control the various events and activities in a specific domain.

To describe the constrained IoT devices, the *Constrained RESTful environment* (CoRE) based on a RESTful architecture is proprosed. CoRE uses *universal resource identifiers* (URI) to identify and discover the resources hosted by constrained nodes. The authors in [224] replace CoRE link format by a semantic-based description, namely JSON-LD, to enable seamless Things to Things interaction in the constrained environment. A light-weight Things management framework residing in an M2M gateway is also proposed.

6.2.2 Web of Things Framework

Guinard *et al.* propose a WoT framework generated a set of REST services to exploit and hierarchically link the sensor node resources via a web interface [230]. This framework speeds up creating ad-hoc applications for end-users. Their work is later applied in the AutoWoT project [231] that aims to rapidly integrate IoT devices into the WoT framework by automatically building the web services to expose the device resources and functions. AutoWot generates these web services based on hierarchical service descriptions created by end-users for the specific devices. In addition, a graphical user interface is also proposed to facilitate creating description process for developers and tech-savvy users.

Another academic framework involving REST principle for integrating IoT devices into the Web is WebPlug [232], which has several functional blocks represented sensor data as the web resources. The users can compose their personal services on physical objects. These services and resources accessibility are enhanced by *Meta-URL* that is proved to deliver considerable benefits in temp of resource discovery and creating mash-up WoT applications. In the same approach, Christophe *et al.* [233] propose a framework for creating and handling IoT devices as virtual objects according to *event-based rule schemas*. The authors in [234] present a service-oriented framework supporting multiple real-time services for Wireless Sensor Networks and smart objects via the Web. SemSense framework [235] aims to collect, process, add semantic meta-data and publish on the Web according to the *linked data standards*. SPITFIRE [35] targets to unlock *the uni-modal closed system* in WoT by generating the semantic sensor description and an effective searching mechanism. After abstracting, the sensors and Things are integrated into *Linked Open Data cloud*. This effort makes the collected data easily accessible on the Web and accelerates the development of WoT applications. With the convergence of academic and commercial worlds, several commercial platforms are emerging with the aim to abstract the heterogeneity of the physical devices by REST web services. There are well-known frameworks like Xively [236], ThingSpeak [237], and ThingWorx [238]. These frameworks support accessing and visualizing the device data from the cloud. Moreover, a RESTful API is offered for the developers to build the custom applications to exploit the collected data.

In the light of this state-of-the-art, there are still missing the semantic description and a lightweight IoT architecture seamlessly presenting and managing the compound objects, which consist of multiple IoT devices and services.

6.3 WoT Asset Description

Before describing WoT Asset Description in detail, we have to identify required components and functions of a Asset:

- The Asset must be able to introduce and discovery itself via self-description meta-data.
- The existing resources and interaction methods must be described in the way that they are fully accessible via Web.
- The devices along with their services belonging to the Assess must be controlled by end-users via defined web services.

Based on requirements outlined above, we introduce the WoT Asses Description (WoT-AD), a semantic description considered as the abstract entry point of an asset. It enables asset resources to be effective discovered, accessed and managed. WoT-AD is an extended concept of the W3C Things Description (W3C-TD), which uses a JSON-LD schema to describe the single Things. The WoT-AD structure consists of three primary sections as illustrated in Figure 6.1: (1) The *descriptive meta-data* describes the general Asset information; (2) The *interaction model* presents the *Asset Resources* (ARs) under semantic schemes; (3) The *Entities* expresses the Things and services constituted of Asset and their relations. More details of each element are described below:

- **Descriptive Meta-data Section:** This section characterizes the Asset by presenting the unique identifier (URI) and general information, such as name, model, description, and link.

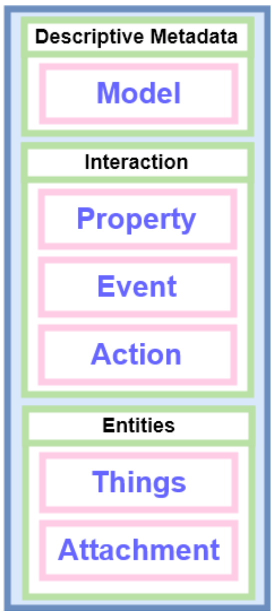


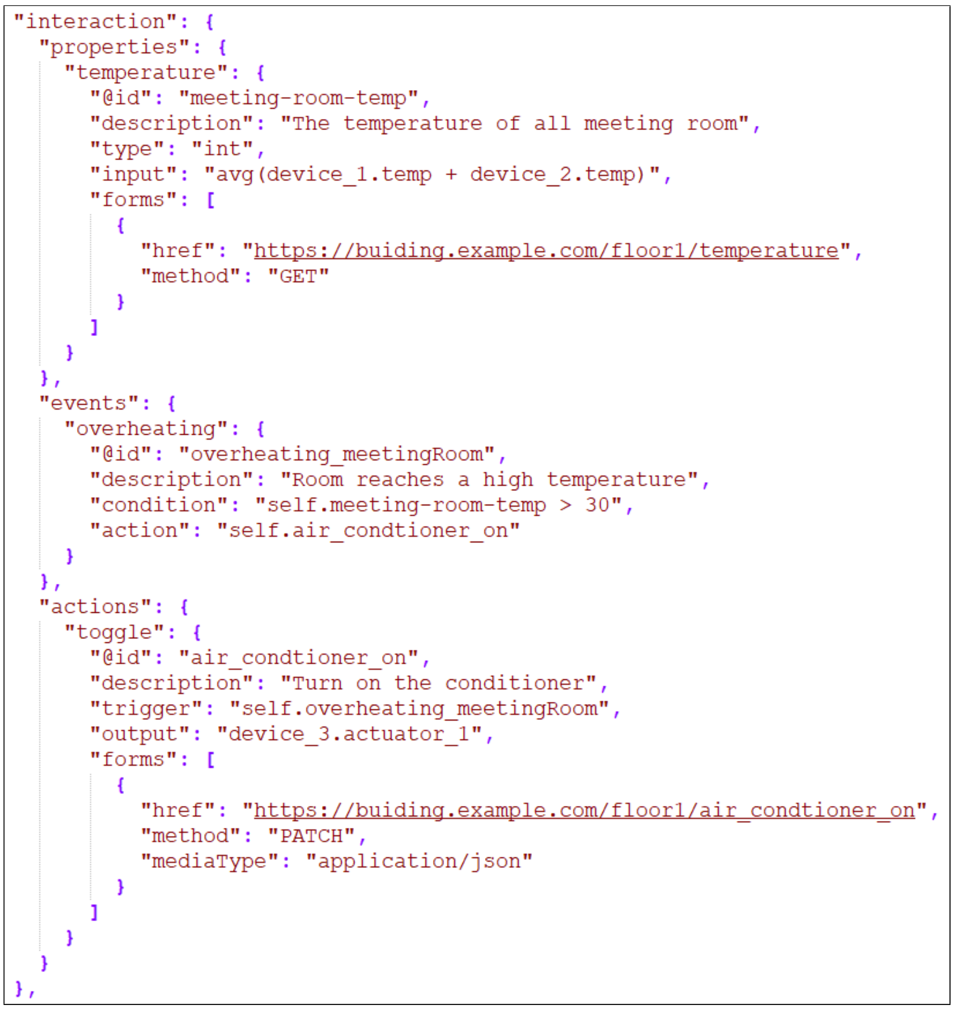
Figure 6.1 – The Asset model overview.

- **Interaction Section:** This section presents the available resources and services of the Asset and their interaction methods. Each resource is an interaction pattern containing two separated parts:

  - Descriptive meta-data presents the general information and configuration of resources, such as type, description.
  - Interaction information presents the connecting information of asset resources, such as connection address, method, and authentication information.

  The interaction pattern is divided into three types: Property, Event, and Action. The Figure 6.2 illustrates the interaction section of a simple WoT-AD that used to abstract a building floor as an Asset.

  - The Property section expresses the state of Asset, which is collected or calculated from the corresponding IoT devices. The property values are timely updated by mathematical formulas complying with *Virtual Sensor Framework*. The end-user could directly access or observe such values via the interaction information defined via URI. As shown in the example, the temperature of the meeting room in the floor is declared as an Asset property and updated from the average temperature collected from *device 1* and *device 2*. An HTTP access API is provided to observe this value.
  - The Event section presents the particular phenomenons detected by a set of specific conditions on Properties. When the condition is satisfied, the corresponding actions are triggered. For example, in Figure 6.2, we identify the overheating event when the value of temperature property is higher than 30 degree. Then, this event triggers an action to turn on the air conditioner.
  - The Action section presents the supported actions of Asset which could be triggered by the corresponding events. The user could also directly access and trigger these actions via the provided services of the Asset. In the Figure 6.2, a turning



```
"interaction": {
  "properties": {
    "temperature": {
      "@id": "meeting-room-temp",
      "description": "The temperature of all meeting room",
      "type": "int",
      "input": "avg(device_1.temp + device_2.temp)",
      "forms": [
        {
          "href": "https://buiding.example.com/floor1/temperature",
          "method": "GET"
        }
      ]
    }
  },
  "events": {
    "overheating": {
      "@id": "overheating_meetingRoom",
      "description": "Room reaches a high temperature",
      "condition": "self.meeting-room-temp > 30",
      "action": "self.air_condtioner_on"
    }
  },
  "actions": {
    "toggle": {
      "@id": "air_condtioner_on",
      "description": "Turn on the conditioner",
      "trigger": "self.overheating_meetingRoom",
      "output": "device_3.actuator_1",
      "forms": [
        {
          "href": "https://buiding.example.com/floor1/air_condtioner_on",
          "method": "PATCH",
          "mediaType": "application/json"
        }
      ]
    }
  }
},
```

Figure 6.2 – The WoT-AD interaction section.

on air conditioner action is trigger by the overheating event, and the command is directly sent to the actuator of *device 3*.

- **Entities and Attachment Section:** This section describes the *Entities* belonging to the Asset. In our proposal, the entity may be a Things or service, namely *Attachment*. Each Thing contains the descriptive meta-data and the address to interact with Things resources and services. The attachments are the cloud-based services, such as open data servers, repositories, device management servers. Such attachments are treated as the IoT Things which have meta-data and interactive address.

6.4 WoT Framework for WoT-AD

6.4.1 Principles and Design

We aim to enable semantic interoperability for the Asset in Massive IoT, by providing a novel concept for Asset description along with a WoT framework. In addition, the proposed framework facilitates the development of IoT applications on the Asset by providing a composition editor. For this reason, the designed architecture complies with following principles:

- High performance and low computational load for the server to handle a massive number of connections.
- Effectively discovering, managing and access the Asset.
- Accelerating the IoT application development on the Asset regardless of the user's skill.
- Handling the heterogeneity of constrained devices in LPWAN context.

In order to archive these principles, we deal with several challenges listing below:

- The connectivity layer of architecture should be able to handle the heterogeneous connections from not only IoT devices but also the open data sources (such as open weather, open MQTT broker). In addition, this layer needs to deal with the heterogeneity in term of LPWAN callback configurations. For example, the SIGFOX forwards the data via HTTP GET method whereas LoRA uses HTTP POST method with different syntaxes.
- The core elements of architecture must automatically discover and index the available Things resources that constitute of the Asset. These resources should be discovered in mash-up applications to facilitate the Asset creation procedure. This part also takes responsibility for updating in real-time the Assess resources from collected data.
- The upper part of architecture must provide simple APIs to allow the user to effectively discover, access, and manage the Asset. Moreover, a graphics editor is necessary to facilitate the Asset creation process.

6.4.2 Overall Architecture

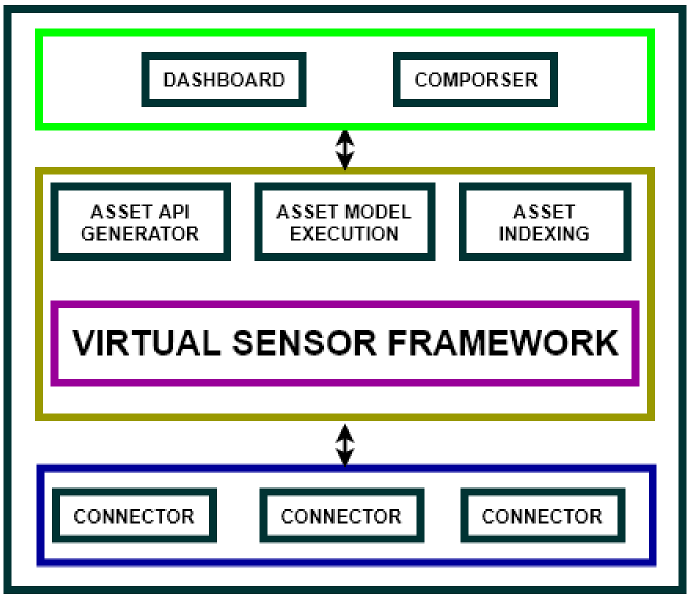


Figure 6.3 – The Asset architecture overview.

Based on the listed principles and challenges, we propose a WoT framework fully exploit WoT-AD concept in Massive IoT scenario. The initial implementation of such architecture is represented by three main components spreading on horizontal layers, whose features properly fit the mentioned principles. The *connector framework* is used at lowest layer [29] to handle the heterogeneous connections from both IoT devices and external data sources. The second component is the *virtual sensor framework* [30] supported producing high-level information from collected data by using virtual sensors. The last component is a graphic editor facilitating the building and configuring process of the Asset by offering the drag-drop actions on HTML5 web interface. As shown in Figure 6.3, the details of proposed architecture are described below:

- **Connection Layer:** This layer handles the connections from heterogeneous devices to the framework through the several connectors. These connectors are directly created and managed by the connector framework [29]. At this layer, collected data from physical devices are aggregated and pre-processed before conveying to upper layers. In case a new device connects to the framework, the its resources are registered and tracked by the sensor tracking service.

- **Processing Layer** This layer is a primary part of our architecture consisted of four main components:

  - *Virtual sensor platform:* It is used to create the Asset by abstracting and linking the device resources together. This platform is also used to update the Asset resources from collected data.

  - *Indexing:* After successfully creating, the Asset resources are indexed by the indexing system. In this way, all the resources are discovered and synchronized with physical devices in real-time.

  - *Asset scripting:* It is used to generate the Asset description and APIs based on defined resources.

  - *Asset model execution:* It coverts the Asset model to a *logical data flow* [30] used by the virtual sensor framework.

- **Presentation layer** This layer contains an interactive HTML 5 web interface, namely *Asset Composer*. This graphical interface supports presenting several IoT components, such as sensor, actuator, Asset, and symbolic link. The end-user could create their own Asset by drag-drop actions. A dashboard to manage the existing Asset also proposed.

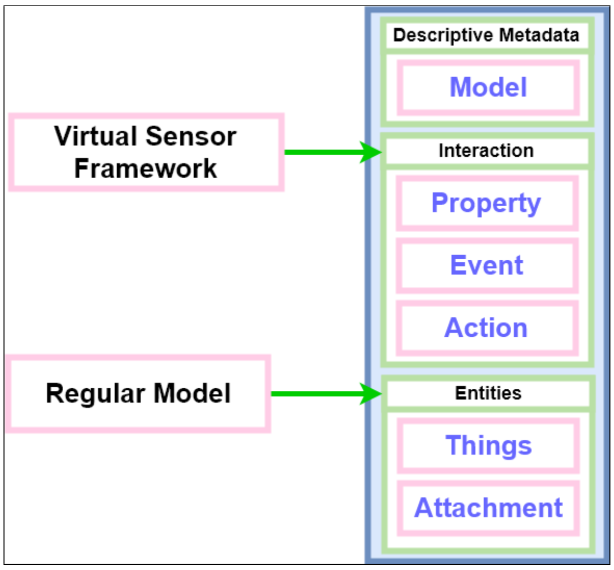


Figure 6.4 – The operation model overview.

The primary requirement of WoT framework is to effectively and timely update and present the available resources. In our proposal, these resources are presented in Asset's property sections and operate as virtual sensors to retrieve and present the collected information from IoT sensors or devices. For this reason, we integrate our framework with a virtual sensor framework [30] which simplifies creating and configuring virtual sensors with the programmable operators (such as rules, formulas, and functions). These virtual sensors are linked together to be a network, namely logical data flow, to produce high-level information from collected data. This information is used to update the property section of the asset. Typically, an Asset is a self-operation object which is periodic updating its property values via an assigned logical data flow. The end-user can configure this frequency in the Asset descriptive meta-data section.

On the other hand, the elements in the Interaction section (properties, events, actions) are build based on complex event processing engine [239]. Therefore, if the current Asset properties reach a certain condition, the corresponding events are inferred and triggered the actions. For example, we present the building floor as an Asset with the properties are the C02 and Humility of each room on the floor. If the CO2 degree of 75% rooms is higher than 1000 ppm, the "suffocating" event is inferred, and the system send instruction commands to relevant actuators to open windows.

6.5 Evaluation

In order to validate the concept of WoT-AD and WoT architecture from the functional point of view, we have utilized the smart space scenario including a Raspberry Pi model B, which is considered as a smart gateway installed our framework. In such context, the Assets are floors consisting of a set of rooms which are monitored by various IoT devices. For example, a monitoring device contains with multiple sensors including Co2, humility, motion detection, temperature, light. All collected information is used to control the air conditioning and light systems for saving energy consumption. The control decisions are based on the whole floor status instead of the single room. Therefore, this context is the best practice to apply Asset.

Following the described scenario, a end-user can visually create and control an Asset via the interface of smart gateway. If the users use the existing Asset template, they only need to assign the IoT device identifications to the template. Based on these identifications, these device resources are discovered and added to the interaction models of WoT-AD. Then, the complete WoT-AD is conveyed to processing layer where (1) the Asset Script component creates the access API for the Asset resources; (2) the Indexing component assigns a unique identity to the Asset for further self-discovery. Based on the composed Asset configure file, the logical-data flow is created to update the Asset properties from the collected data. This ensures the Asset data timely updated and reacted with the environment changes via configured events and actions in Asset.

The software environment along with a light-weight local database for storing Asset configuration consume around to 100MB of spaces on the Raspberry Pi comparing with 32GB memory card. The consumed operations of the framework are collecting, processing and updating the Asset information in processing layer. During the operational cycle mentioned in the scenario, the CPU loads from 20 to 30 percent. The high performance and scalability of such operation are ensured by Virtual Sensor Framework [30].

6.6 Conclusion

In a nutshell, this work presents a semantic description for the Asset enabling semantic interoperability in Massive IoT context. A WoT architecture is also defined and implemented to fully exploit the proposed concept. The architecture supports the interaction with various IoT devices and sensors based on the connector model at the lowest layer. The core element of the architecture is a Virtual Sensor Framework timely updated the Asset resources from collected data. At the top layer, we implemented a graphics editor supported the Asset description composing. The effectiveness of the designed solution is ensured by choosing and combining some technologies and IoT frameworks that have been practically demonstrated in real use-cases. To improve and extend the current works, we will evaluate and optimize WoT-AD in various IoT scenarios.

Part III

Reliability in IoT

7 An Active Learning Method for Errors and Events Detection

Anomalies are pervasive in time series data, especially IoT sensor readings. Existing methods for anomaly detection cannot distinguish between *anomalies* that represent data errors, such as incorrect sensor readings, and notable *events*, such as the watering action in soil monitoring. In addition, the quality performance of such detection methods highly depends on the configuration parameters, which are dataset specific. In this work, we exploit active learning to detect both errors and events in a single solution that aims at minimizing user interaction. For this joint detection, we introduce a non-parametric algorithm, which accurately detects and labels anomalies with a novel concept of neighborhood, and unsupervised probabilistic classification. Given a desired quality, the confidence of the classification is then used as termination condition for the active learning algorithm. Experiments on real and synthetic datasets demonstrate that our approach achieves high F-score by labeling a very limited number of data points, such as 2 to 10 points in the Yahoo! datasets to obtain 80% accuracy on average. We also show the superiority of our solution compared to the state-of-the-art approaches and the impact of our methods over datasets with increasing percentages of errors and events.

7.1 Introduction

Anomaly detection is an important tasks in several domains, such as intrusion detection systems, financial fraud detection, and Internet of Thing (IoT). It has been estimated that collected data could have from 2.3% to 26.9% error rate [240]. Applications built upon imprecise time series can potentially result in losses in the millions of dollars to businesses [241]. As a concrete example, in forest fire detection many sensors are deployed to monitor the concentration of carbon-monoxide and various organic compounds [242]. Potential problems are detected before occurring by combining collected data with the external weather information (e.g., wind speed, temperature, humidity). Imprecise detection coming from abnormal data could significantly decrease the system reliability and results for remedial works. The efficacy of such response systems highly depends on the performance of the anomaly detection algorithms [243].

Anomaly detection over time series is often applied to filter out dirty data. This means that the detected points are discarded as noise. Unfortunately, the eliminated data may contain notable events, also known as *change points*. These changes occur by accident (e.g., a fire in a forest) or because of human intervention (e.g., watering the tree). Preserving these events is essential to interpret the context. For example, to optimize the watering schedule,

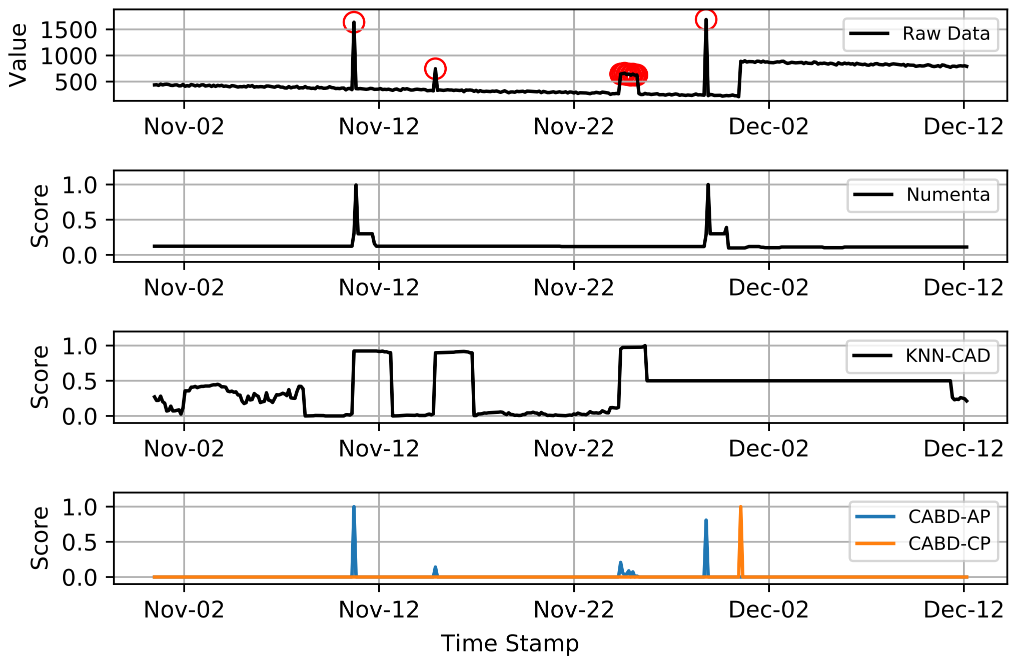


Figure 7.1 – An example of IoT data (top plot) and detection results for four algorithms.

a city environment management company deploys the sensors to monitor the impact of watering on soil humility. However, a significant increase of soil humility due to water can be detected as anomaly and removed from the data [162]. Thus, the ability to explicitly distinguish between anomalies and events is needed.

The first plot from the top in Figure 7.1 visually presents this challenge in real ultrasonic sensor data that we obtained from an IoT solution company. The sensor is plugged on the top of a tank to monitor its liquid level (y axis) over time (x axis). As shown in the figure, some sudden changes appear, either in isolation (at 08-November and 14-November) or as small groups (at the time from 24-November to 26-November). These abnormal values are sensor errors, and should be fixed or removed from the data set. On the other hand, the data change reflecting the filling of the tank (at 30-November) should be preserved.

The most common anomaly detection methods on time series use traditional statistical methods, e.g., neighbor-based [81, 125, 126, 127], ensembles [244, 245], and probabilistic models [246, 247]. They consider a data point as abnormal if it significantly differs from historical observations. Unfortunately, change points also show this behaviour in practice and existing detection methods recognize such change points as anomaly. For state-of-the-art anomaly detection methods, such as Numenta [247] and KNN-CAD [246], the presence of change points in time series significantly decrease the quality of the detection. This often leads to skip other true anomalies. The basic idea of Numenta and KNN-CAD is to use a clean data set as training set $(x_1, \ldots, x_m)$ from historical data to predict the next value of x_{m+1}. Then, they compute a measure of prediction error. In a final step, they use a probabilistic model to estimate the anomalous state. Because the change points are not labelled in the training data, their presence in examining data directly affects the prediction result. There are two cases: (1) If these algorithms incorrectly predict an abnormal point as a change point, this point could be ignored because the prediction error is minor; (2) If these algorithms incorrectly classify a change point as an abnormal point, all the points after the change point are detected as anomaly points. As shown in Figure 7.1, Numenta algorithm ignores all collective anomalies. The KNN-CAD fails in all its detection for this dataset. In addition, the performance of such detection highly depends on parameter configuration that is data specific. For example, KNN-CAD algorithm requires a "window length" parameter defined the size of sliding window. This parameter varies with different datasets. Owing these limitations, the detection result is very poor (f-score is below 20%), as illustrated in Figure 7.1.

Since automatic anomaly detection algorithms do not work well without discrimination between anomalies and change points, one approach is to use supervised learning methods to model such distinction [248, 249, 250]. However, labeled data is not available in general. This motivates our idea to exploit an *interactive* approach. The goal is to minimize the user involvement while guaranteeing quality over the results of the process. To achieve this goal we design an active learning method to obtain the truth from users. Our experiments show that labeling a very limited number of data points can significantly increase the detection quality.

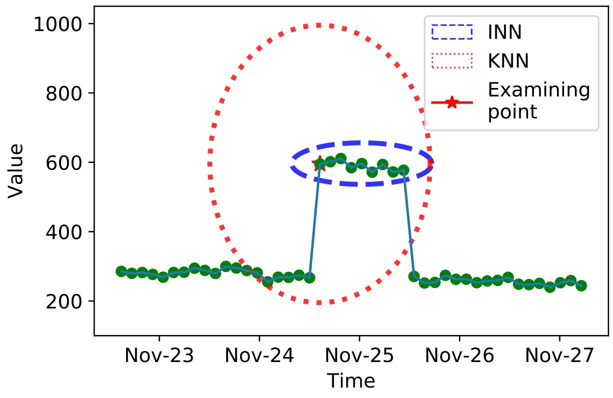


Figure 7.2 – Comparing Inverse Nearest Neighbor (INN) and K-Nearest Neighbor (KNN).

To bootstrap the process and reduce the number of interactions, we propose a non-parametric algorithm that accurately detects both anomalies and change points based on the combination of a novel concept of neighborhood, namely *Inverse Nearest Neighbor* (INN), and unsupervised probabilistic classification. The idea of INN derives from observing that two objects have a close relationship if they have a bi-directional connection. Applying this concept to data object, object A and object B have strong relation if A is a top-k neighbor of B and vice versa, B is a top-k neighbor of A, for any k. Leveraging the definition of INN, the type of a data point (anomaly or change) can be classified through evaluating as set of scores modelling INNs properties. The benefit of the INN concept is also demonstrated in detecting collective anomalies. If a data point is identified as abnormal, its INNs is highly anomalous. Our algorithm propagates the anomaly score to such INN to expose the whole anomaly pattern. In addition, the superiority of INN in comparison to existing neighborhood concepts such as k-nearest neighbor proposed in [251, 252, 253] is that the number of neighbors for each data point are not necessarily identical and are not pre-defined with a parameter. In fact, the search algorithm of INN is non-parametric whereas the searching algorithm of K-Nearest Neighbor (KNN) requires a data-specific parameter, namely "K-distance" [147]. Finding correct K-distance value for each dataset is extremely hard. Figure 7.2 illustrates the difference of INN and KNN. When evaluating if a data point belongs to a collective anomaly, its INN contains entirely such anomaly while KNN with the inappropriate K parameter contains both anomalous and normal points. We discuss INN in more detail in the next section.

Depending on the use-cases, a user requires different detection quality values. For example, a fleet management application requires high accuracy in discriminating sensor errors and events, while applications monitoring CO2 or temperature values do not require such level of accuracy. This requirement leads to a new challenge: how to satisfy the user's desired quality while minimizing the user interaction. To address this challenge, we let the users express a desired *minimum confidence* over the data that we are processing. The confidence of a classification model is then used as a termination condition for an active learning process, according to the user input. More accuracy demands more points labeled to enrich the model until the desired confidence is achieved. Experiments demonstrate that higher confidence requirements lead an increase in the accuracy for the error and event detection.

Our contributions in this work are summarized as follows:

- The *comprehensive anomaly and change point detection* algorithm (CABD), a novel non-parametric method for detecting both errors (i.e., single and collective anomalies) and events (i.e., breakpoints).
- The novel concept of *inverse nearest neighbor* (INN) and active learning using uncertainty sampling scheme are applied to improve CABD effectiveness and efficiency.
- We have implemented CABD as a Python library and extensively tested it in a production environment. The prototype produces high-quality detection in practical IoT use-cases.

The remainder of this chapter is organized as follows. In Sections 7.2 and 7.3, we formalize the problem of anomaly detection and related definitions. The INN concept and CABD algorithms are presented in Section 7.5 and Section 7.4, respectively. Section 7.6 reports the quality evaluation of our method through two real use-cases. Section 7.7 discusses related work, and conclusions and future work are reported in Section 7.8.

7.2 Preliminaries

7.2.1 Related Definitions

In the experiments in this chapter (Section 7.6), we use the *Euclidean distance* to calculate the distance of two data points in the INN concept. The Euclidean distance is the straight-line distance between two points in Euclidean space [254]. It is calculated by the root of square of the differences between coordinates of two points.

Definition 1. The Euclidean distance between data point p = (p_1, p_2) and q = (q_1, q_2) is given by

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2} \tag{7.1}$$

In calculus, the second derivative represents the rate of changes of a data point. It is widely used to identify the critical points (e.g., the local minimum or local maximum) [255]. In our solution, we use the second derivative to find the anomaly candidate in Section 7.5.2.

Definition 2. The absolute value of second difference of $x_i(p)$, denoted as $\triangle'' x_i$, which is defined that:

$$\triangle'' x_i(p) = |\triangle x_i - \triangle x_{i-1}|,\ i = 1, 2, 3, ..., n \tag{7.2}$$

While $\triangle x_i$ is the absolute value of first difference of $x_i \in X$, is defined that:

$$\triangle x_i(p) = |x_i(p) - x_{i-1}(p)|,\ i = 1, 2, 3, ..., n \tag{7.3}$$

In general, Symbolic Aggregate Approximation (SAX) algorithm transforms a time series into a strings [256]. This algorithm is used to detect the unusual pattern in time series [257], the duplicated shapes in large databases [258], and the time series motifs [259]. We use SAX to calculate a score in our score metric that represents the frequency of occurrence of a data pattern in time series in Section 7.5.3.

Definition 3. *Piecewise Aggregate Approximation* (PAA) transforms a time-series X of length n into vector $\bar{X} = (\bar{x}_1, ..., \bar{x}_M)$ with $M \leq n$ where:

$$\bar{x}_i = \frac{M}{n} \sum_{j=n/M(i-1)+1}^{(n/M)i} x_j \tag{7.4}$$

Definition 4. *Symbolic Aggregate Approximation* (SAX) transforms a time-series X of length n into an arbitrary string by using PAA. A time series X length n can be represented by a word $\hat{X} = \{\hat{x_1}, \hat{x_2}, ..., \hat{x_n}\}$ with $\hat{x_i}$ is a character of alphabet. Let denoted α_j is the j^{th} element of the alphabet. θ_{j-1}, θ_j are given thresholds.

$$\hat{x_i} = \alpha_j \quad s.t \quad \theta_{j-1} \leq PAA(x_i) \leq \theta_j \tag{7.5}$$

Standardization is a necessary step to deal with differing scales in input values. Standardizing a dataset is to rescale the data distribution so that the overall mean and standard deviation are 0 and 1, respectively. A value x_i in time series data X is standardized as follows:

$$x_i = \frac{x_i - mean(X)}{std(X)} \tag{7.6}$$

7.2.2 Anomaly Types

Anomaly detection is a technique used to identify unusual patterns that do not conform to expected behaviors, also called outliers. For this activity there are many applications, from intrusion detection to system monitoring. It is important to establish some boundaries on the definition of an anomaly. Anomalies can be broadly categorized as [81]:

- **Point anomalies**: A single instance of data is anomalous if it is significantly different from the remaining data.
- **Contextual anomalies:** The abnormality is context specific. This type of anomaly is common in time-series data. For example: a reported 30 Celsius degrees temperature in summer is normal but may be abnormal in winter.
- **Collective anomalies:** This anomaly type contains a set of consecutive point anomalies represented as an abnormal data *pattern.* This pattern does not comply with the dataset distribution.

7.2.3 Break Point

In the simplest form, a break point, also called a change point, is the point at which the statistical properties of a sequence of observations change [260]. Break point detection is applied in vary application areas from finance, environment, health care to industrial maintenance [261, 262, 263]. More formally, let assume we have a time series $X = \{x_1, x_2, ..., x_n\}$ which has m break points at the position $\mathcal{C} = \{c_1, c_2, ..., c_m\}$ with $c_m < n$. The break points separate the data set into $m + i$ segments such that the statistical properties of i^{th} segment $\{x_{c_{i-1}}, ..., x_{c_i}\}$ and $(i+1)^{th}$ segment $\{x_{c_i}, ..., x_{c_{i+1}}\}$ are different in some way.

7.3 Problem Statement

We consider a time series $X = \{x_1, x_2, ..., x_n\}$ of n observations, where x_i is the i^{th} data point, that may contain both errors and events. Its errors could be either single or collective anomalies.There are two main observations with reviewed algorithms, such as LOF, Numenta, and KNN-CAD. First, a change point in X is usually detected as anomaly and simply discarded. Second, the anomaly detection performance highly depends on configuration parameters which are data specific. For instance, LoF algorithm is based on the KNN concept and requires the number of nearest neighbors (K parameter), while KNN-CAD requires the size of the sliding window. Moreover, the presence of change points also reduces such performance. Lastly, labeling anomaly data for training sets requires expensive manual labour (such as Isolation Forest [264], One-class Support Vector Machines [265], Robust Co-variance [266]).
Let $ac_i = \{x_i, \ldots, x_{i+s} \mid x \in X, s \in \mathbb{N}\}$, as_i and c_i denote a collective anomaly sized s, a single anomaly, and a change point at data point $x_i \in X$, respectively. Our problem statement is formalized as follow:

Problem: *Given a desired detection confidence q and time series $X = \{x_1, x_2, ..., x_n\}$, detect any collective anomaly ac_i, single anomaly as_i, and change point c_i with confidence above q while minimizing user interaction.*

Example 1. *Consider time series X in Figure 7.1, part of real IoT sensor data, with a collective anomaly occurring around Nov-24 and three single anomalies at Nov-10, Nov-14 and Nov-30. This time series also contains a change point at Dec-01. As shown in Figure 7.1, all reviewed detection algorithms cannot correctly detect the abnormality. For example, Numenta cannot detect the sequence of errors and confuses change points with abnormal points. The detection result of KNN-CAD is even worse than Numenta. All single anomalies are incorrectly detected as collective anomalies. Our goal is to effectively detect both various kind of anomalies and change points in a single algorithm with minimal input from the user.*

7.4 Inverse Nearest Neighbor

We call *r class* of a point p the group of points induced by the r-nearest neighbor for p. Two points are in a Inverse Nearest Neighbor (INN) if one belongs to the r class of the other and vice versa, for any r value and $r \in \mathbb{N}$. More precisely, let $NN_r(x_i)$ denote the set of r nearest neighbors for data point x_i. Note that the number of objects in $NN_r(x_i)$ equals r. Given the time series $X = \{x_1, x_2, ..., x_n\}$ and two points $x_m, x_i \in X$, if point x_m belongs to the r nearest neighbors of point x_i and the point x_i belongs to the r nearest neighbors of point x_m, then x_m is an *inverse nearest neighbor* of x_i at $r - distance$, denoted as:

$$INN_r(x_i) = x_m \textit{ iff } \begin{cases} x_m \in NN_r(x_i) \\ x_i \in NN_r(x_m) \end{cases} \tag{7.7}$$

The major difference of INN with existing neighborhood concepts such as k-nearest neighbor [251, 252, 253] is that INN is that the number of neighbors for each data point are not necessarily identical and pre-defined. In this way, the search algorithm of INN is non-parametric. It starts from the nearest neighbor (top 1 nearest neighbor) of examining point and stop when it cannot find more INNs. The detail of this algorithm is described in Algorithm 1. In addition, we apply *KD-tree* [267] technique to enhance searching performance. The more detail of searching steps are illustrated in Example 2. In the worst-case scenario, the INN of a data point is the whole dataset if this dataset distribution is a flat line. Hence, a optimized version is discussed in Section 7.5.5.

Algorithm 1 INN Searching of data point X_i

```
Input:  kd-tree of time series X and data point $X_i$
Output: List of INN's $X_i$
1. Initializing: flag = 0, r = 1, $INN(X_i) = \varnothing$
2. Use kdtree to find the $r$ nearest neighbors $Y$ for $X_i$
      || Find the $r$ nearest neighbors for each $Y_i \in Y$
      || If $X_i \in NN(Y_i)$ and $Y_i \notin INN(X_i)$ then
      ||         $INN(X_i) = INN(X_i) \cup (Y_i, r)$
3. Compute the size $INN(X_i)$
      If this size does not changed
            Return $INN(X_i)$
      Else
            r++
            go to step 2
```

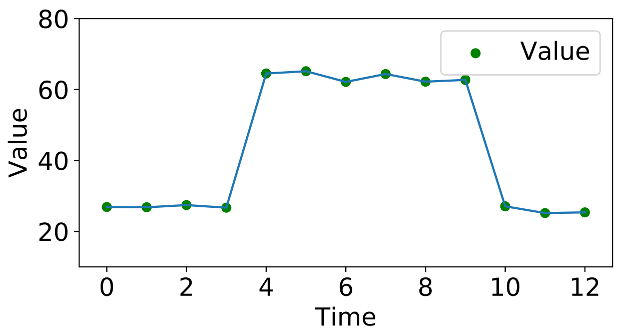


Figure 7.3 – An Example of INN.

Example 2. Consider the time series X = {26.9, 26.8, 27.4, 26.7, 64.5, 65.1, 62.1, 64.4, 62.2, 62.7, 27.1, 25.2, 25.4} with thirteen data points presented in Figure 7.3[1]. X has a

[1]This example only aims to illustrate the INN searching process. It is different with evaluated experiments in which all datasets is standardized by using the Equation 7.6.

collective anomaly on six points from x_4 to x_9. Assume we would like to find the INN of x_4. The algorithm starts at r = 1, we have $NN_1(x_4) = \{x_5\}$ and $NN_1(x_5) = \{x_4\}$. Referring to Equation 7.7, x_4 and x_5 are INN at distance 1. Similarly, with r values from 2 to 5, we always identify $\{x_6, \dots, x_9\}$ belonging to the INN of x_4. This is because with $r = 5$, $NN_5(x_9) = \{x_8, x_7, x_6, x_5, x_4\}$ still contains x_4.
To describe $r = 6$, for simplicity, we use Euclidean Distance to calculate the distance between data points. With $d(x_4, X)$ =[37.9, 37.8, 37.1, 37.0, 0.0, 1.2, 3.1, 3.0, 4.6, 5.3, 37.4, 39.8, 39.9], we have $NN_6(x_4) = \{x_5, \dots, x_9, x_3\}$. Because of $\{x_5, \dots, x_9\} \in INN(x_4)$ (they are processed at r values from 1 to 5), we examinate x_3. Using Euclidean Distance, we calculate $d(x_3, X)$ = [3.0, 2.0, 1.2, 0.0, 37.0, 38.5, 35.6, 37.9, 35.8, 36.5, 7.0, 8.1, 9.1]. Based on these values, we have $NN_6(x_3) = \{x_0, x_1, x_2, x_{10}, x_{11}, x_{12}\}$. As we see, $x_3 \in NN_6(x_4)$ but $x_4 \notin NN_6(x_3)$. Therefore, x_3 does not belongs to the INN of x_4. The INN searching for x_4 is stopped at $r = 5$ and the $INN(x_4) = \{x_5, x_6, x_7, x_8, x_9\}$ [2].

7.5 Detection using Active Learning

Unlike the existing anomaly detection algorithms that only target detecting either abnormal or change points, our goal is to effectively detect both anomalies and change points in a single algorithm with minimal input from the user. In this section, we first present the overall algorithm with active learning. Then, we briefly explain each step along with related definitions.

7.5.1 Algorithm Overview

Let $X = \{x_1, x_2, ..., x_n\}$ denote a time series, where Y and Z are set of anomaly points and change points of X, respectively. Algorithm 2 presents the major steps of our proposal, which takes X as an input and produces Y, Z, and their *confidence Weights* (CW). In addition, the algorithm allows the user to configure the minimum desired confidence to ensure detection quality. The major steps are described as below:

1. **Candidate Estimation**, in Line 1, generates potential candidates, denoted by θ, from extreme values in time series based on their absolute second derivative (Definition 2).

2. **Score Computation**, in Line 3, computes a score metric from INN for each candidate x_i in θ. This metric includes magnitude score, correlation score, and variance score, denoted as $\beta^{(x_i)}$.

3. **Score Evaluation**, in Line 4, uses a probabilistic classification to classify the candidates into three classes including change points, anomaly points, and normal points based on their score metric. Active learning using the uncertainty model, described in Equation 7.13, is applied to minimize user interaction. The most uncertain points are queried and labeled by the users. The output of this step is the CW (denoted as $\mathcal{C}^{(x_i)}$).

4. **Classification Evaluation**, in Line 5, triggers the active learning process if the confidence weight of the candidate is lower than the user's desired quality.

[2] In next sections, the INN of a data point means the set of points acquired from INN Searching Algorithm (Algorithm 1) on that point.

Algorithm 2 Anomaly and Change Point Detection

```
**Input:** Time series X, User's desired confidence $\gamma$
**Output:** Error list $Y$, Change point list $Z$
1. $\theta \leftarrow$ Candidate(X)
2. Y, Z = [ ]
3. For $x_i$ in $\theta$ do
       $\beta^{(x_i)} \leftarrow$ Score($x_i, X$)
4. $\mathcal{C}, Y, Z \leftarrow$ Evaluate Detection($\beta$)
5. If $\min(\mathcal{C}) \leq \gamma$ then
       Labeling and Go to step 4

   Return $Y, Z$
```

7.5.2 Anomaly Candidate Estimation

Our goal is to recognize both of errors and events. Therefore, we first introduce a method to identify the critical changes which may contain the notable behaviors (errors or events).

Standard measures such as mean, variance, and correlation are commonly used in change point detection algorithms [268]. In our algorithm, we identify the change of a data point in time series based on its absolute second derivative, namely *change score*. Formally, given time series X = $x_1, x_2, ..., x_n$. The change score of X is denoted by ∂:

$$\partial(X) = \{\triangle"x_1, \triangle"x_2, ..., \triangle"x_i\} \,|\, i \in \{1, 2, ..., n-1\} \tag{7.8}$$

To identify the candidates, we use *Median Absolute Deviation* (MAD) concept which is a robust measure of the variability in data [269]. Following [270], MAD is more resilient to outliers than the standard deviation. If MAD of the change score of a data point is higher than MAD of the change score of the whole data set, it is considered to be practically abnormal candidate. We validate these candidates in the latter detection steps.

Definition 5. Given time series X and the change score ∂, MAD is defined as the median from sample median.

$$MAD(X) = median(|\partial(X_i) - median(\partial(X))) \,|\, X_i \in X \tag{7.9}$$

7.5.3 Score Computation

In this step, we compute the score metric of each candidate calculated in previous step. There are three scores in this metric: (1) *Magnitude score*; (2) *Correlation score*; (3) *Variance score*. Each score represents a characteristic of the candidate. In more detail, the magnitude score of a candidate describes the ratio of its INN size over the dataset size. Based on the anomaly definition in [81], the size of a anomaly pattern (a collective anomaly) must be less than five percent of dataset. Therefore, if the magnitude score is higher than five percent, the candidate may be a normal point. Similarly, the correlation score represents the regularity of its INN pattern. If this pattern rarely occurs, the candidate may be a abnormal point. The variance score represents the changes of the standard deviation after removing the INN of the candidate. If this score is extremely low (around 0), the candidate is highly normal. Let SS(x_i) be the INN size of data point x_i and SPa(x_i) be the INN of x_i including SS(x_i) numbers of adjacent points in both sides. The details of the three scores are described below:

Definition 6. Magnitude score (MS) of data point is the ratio of its INN size over the size of dataset, denoted by MS. Given time series $X = \{x_1, x_2, ..., x_n\}$ length n and $x_i \in X$, the MS of x_i is defined as:

$$MS(x_i) = \frac{size(INN(x_i))}{n} \tag{7.10}$$

Definition 7. Correlation Score (CS) of a data point is the frequency of occurrence of its INN pattern, which is represented as a string by using SAX, in the dataset. (definition 4). Given time series $X = \{x_1, x_2, ..., x_n\}$ and $x_i \in X$

$$CS(x_i) = frequency\left(\frac{SAX(INN(x_i))}{SAX(X)}\right) \tag{7.11}$$

Definition 8. Variance Score (VS) describes the change of standard deviation of spreading pattern with k-neighbors after remove spreading pattern.

$$VS(x_i) = \frac{std(SPa(x_i) - INN(x_i))}{std(SPa(x_i))} \tag{7.12}$$

The algorithm 3 illustrates the score metric calculation process. These scores are calculated in parallel to optimize performance. At final step (Line 5), all scores are collected and formed as a feature input (a three dimension matrix) based on the classification model.

Algorithm 3 Score Computation

```
Input:  Data point $x_i$, time series X
Output: Score Metric $\beta^{(x_i)}$
1. $\eta \leftarrow$ SS($x_i$, $X$)
2. $\kappa \leftarrow$ MS($x_i$, $\eta$)
3. $\xi \leftarrow$ CS($x_i$, $\eta$)
4. $\varphi \leftarrow$ VS($x_i$, $\eta$)
5. $\beta^{(x_i)} \leftarrow [[\kappa], [\xi], [\varphi]]$
Return $\beta^{(x_i)}$
```

7.5.4 Score Evaluation

Relying on the metric scores, the Score Evaluation step uses a probabilistic classification algorithm to estimate the probability of data point x_i to be an anomaly point, change point, or normal point. CABD is designed with high modularity and flexibility. It allows to plug-and-play different classification algorithms (e.g., random forest [271], gaussian process [272], AdaBoost [273]). By default, CABD uses the *random forest classification*, which is tolerant with over-fitting and suitable to varied datasets [271]. Initially, without user intervention, the classification works on a set of initiated hypotheses. With the presence of human, the existing active learning using uncertainty sampling scheme [274], namely *CAL*, is directly applied to increase classification performance by labeling the most uncertain instances.

In CAL, we examine the most likely class of data points $x_i \in X$ based upon the probability produced by the classification algorithm. This probability is also considered as the confidence weight of the examining data point. We decide whether to require its label $y_i \in Y =$ {abnormal point, normal point, change point} from the users based on the uncertainty of the classification result defined by:

$$\mathcal{U}(x) = 1 - P(\hat{x}|x) \tag{7.13}$$

where x is the data point and $\hat{x}$ is the most likely classification. The querying process of CAL is stopped if all confidence weights are higher than the user's desired confidence. By default, this confidence value equals 0.8.

Example 3. If a data point x_i can be classified across the three labels (abnormal point, normal point, change point) with confidences $[0.1, 0.3, 0.6]$, it is considered a change point with 0.6 confidence weight and 0.4 uncertainty.

To ensure that our solution could correctly detect anomaly and change points in case of lacking of user interaction[3], the initial training set of the probabilistic classifier is build on a set of hypotheses $\mathcal{H}$, which include three main decision rules based on the score metric:

1. The magnitude score of an abnormal point must be lower than k%. This means, the spreading pattern size of this point is lower than k% of data size. Particularly, the spreading pattern size of single anomaly equals 0.
2. The correlation score of an abnormal point must be lower than c%. This means, the spreading pattern of this point must occur lower that c% frequently in dataset
3. The variance score of an abnormal point must be higher than v%. This means, the standard deviation of spreading pattern with k-neighbors must reduce at least v% after removing the spreading pattern.

From observing the properties of change points and various anomaly types in the practical dataset, we derive the set of threshold $[k, c, v]$ as 0.05, 0.1 and 0.5 respectively. Given the set of examining data points X, label Y = {abnormal point, collective anomaly, change point, normal point}, threshold $\theta = [0.05, 0.1, 0.5]$, we set the hypotheses $\mathcal{H}$ as follows:

$$\left\{ \begin{array}{llll} h_1 & : & \textbf{Anomaly Point} & \text{if} \left| \begin{array}{l} SS(x_1) = 0 \\ \beta_\xi^{x_i} \leq 0.1 \\ \beta_\varphi^{x_i} \geq 0.5 \end{array} \right. \\ \\ h_2 & : & \textbf{Anomaly Pattern} & \text{if} \left| \begin{array}{l} \beta_\kappa^{x_i} \leq 0.05 \\ \beta_\xi^{x_i} \leq 0.1 \\ \beta_\varphi^{x_i} \geq 0.5 \end{array} \right. \\ \\ h_3 & : & \textbf{Change Point} & \text{if} \left| \begin{array}{l} \beta_\kappa^{x_i} > 0.05 \\ \beta_\xi^{x_i} > 0.1 \\ \beta_\varphi^{x_i} < 0.5 \end{array} \right. \\ \\ h_4 & : & \textbf{Normal Point} & \text{if} \quad \notin \{h_1, h_2, h_3\} \end{array} \right\} \tag{7.14}$$

[3] All our experiments in Section 7.6 are separated into "with and without active learning" cases.

The algorithm 4 summaries the CAL of active learning in Score evaluation step. Let denote the κ and φ be the confidence weight and uncertainty, respectively.

Algorithm 4 Score Evaluation

```
Input: Unlabeled data set $X$, probabilistic model $Z$,
initial training set $\mathcal{V}$, threshold $\gamma$.
Output: $[\kappa, \varphi]$
1. $[\kappa, \varphi]$ = Z(X,$\mathcal{V}$)
2. While $min(\kappa) \leq \gamma$ do
       $x_t \leftarrow Query(x_t, \varphi, X)$
       Label $y_t$ for $x_t$
       Set $\mathcal{V} = \mathcal{V} \cup \{(x_t, y_t)\}$
       Update $[\kappa, \varphi]$ = Z(X,$\mathcal{V}$)
Return $[\kappa, \varphi]$
```

7.5.5 Complexity Optimization

Among the major steps of the proposed algorithm, the score calculation step is optimizable in searching INN of the candidates. First, we identify that INN searching cost could be pruned by applying a binary search method which reduces the searching complexity from O(n) to O(Log n). Moreover, we add a new the stopping condition of INN searching based on maximum size of INN.

Intuition: Recall that when searching the INN of data point x denoted INN(x) in Algorithm 1, the n value denoted the size if INN(x) starts at one and increases by one until n is not change. The complexity of such approach is O(n) with n is the size of INN(x). This could be optimized by using binary searching method to find the INN set for both sides (left and right side) of the data point x. The complete INN is the union of two sets. Thereby, the complexity is reduced from O(n) to 2*O(Log $\frac{n}{2}$). The mandatory parameter of the binary search is the maximum searching position. In practice, if the size of an abnormal pattern is higher than five percentages of data set, it could not be considered a collective anomaly. Thus, this boundary could be used as the maximum searching range.

Given data point x_i, searching range threshold t. The algorithm 5 illustrates the Binary INN searching for the right side of data point x_i.

Algorithm 5 Binary INN searching

```
Input: Data point $x_i$, threshold t
Output: $INN_R(x_i)$
1. $L = i; R = t - 1; INN_R(x_i) = []$;
2. While $L \leq R$ do
       $m = floor((L + R)/2)$
       If $x_m \in KNN(x_i)$ and $x_i \in KNN(x_m)$
           $L = m + 1$
       Else
           $R = m - 1$
4. $INN_R(x_i) = [x_i, \ldots, x_m]$
5. Return $INN_R(x_i)$
```

7.6 Evaluation

In this section, we experimentally evaluate the quality and efficiency of our proposal on both real and synthetic datasets. These datasets are standardized by method defined in Section 7.2.1 before evaluating. Such results are also compared with reviewed anomaly detection approaches. Our goal is to demonstrate that:

- The superiority of CABD w.r.t existing algorithms in both detection quality (anomaly, change point detection) and runtime over real and synthetic datasets.
- The effectiveness of INN concept and active learning in our proposal.
- CABD could be a complementary part to enhance data repairing algorithms.

7.6.1 Metrics of Measurement

To evaluate the efficiency of our proposal, we use three measuring metrics including Precision, Recall and F-score. Let S denote the number of the anomaly or change points detected by the algorithm and G denote their ground truth. The precision (P) and recall (R) are defined as:

$$Precision = \frac{|S| \cap G}{|S|} \tag{7.15}$$

and

$$Recall = \frac{|S| \cap G}{|G|} \tag{7.16}$$

respectively. F-score or F-measure, a simple way to balance the P and R of an overall detection result, is defined as:

$$F - score = 2 * \frac{P * R}{P + G} \tag{7.17}$$

To assess the advantages of using interactive learning in comparison to manually labeling all cases, we use a benefit function calculated from the ratio of the number of human actions (labeling) over the total number of anomaly and change points [275]. Formally, given the number of queries by active learning T_A and the total number of anomaly and change points M, we defined the benefit function as:

$$BNF = 1 - \frac{T_A}{M} \tag{7.18}$$

7.6.2 Benchmark Datasets

7.6.2.1 Real dataset

IoT data with real errors: The data ($https://github.com/kimhungGCZ/anomaly\_dataset$) is collected from 2 real ultrasonic sensors deployed on the top of tanks to monitor liquid levels. Errors naturally occur without any human interactions. Since we entirely manage the tank operations such as filling or consuming, these errors and change points are manually labeled as ground truth.

Yahoo data with real error: The yahoo lab data ($http://labs.yahoo.com/Academic\_Relations$) provides a number of datasets taken from real production traffic to some Yahoo's properties. The abnormal points are marked by humans so they are probably not consistent. In addition, the change points are not presented. Thus, these datasets are best used to measure the recall factor of anomaly detection.

7.6.2.2 Synthetic dataset

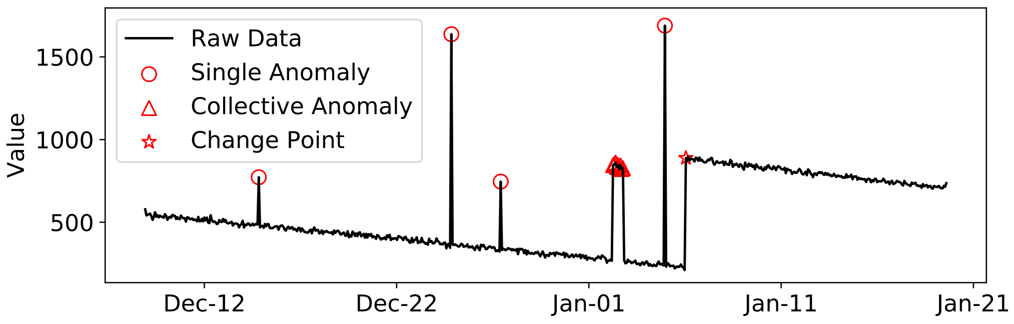


Figure 7.4 – An example of Synthetic datasets.

Synthetic datasets aim to assess the efficacy of our algorithm in the presence of various anomaly types (local, global, single, and collective anomalies) and change points in different proportion. To create these datasets, we first generate the data points following real data distribution. Then we fit this data to a time series obtained from the production environment to preserve the trend and seasonality. Lastly, we randomly inject a mix of anomaly types with varied lengths and magnitudes. These anomalies also recorded to evaluate our proposed algorithm. Figure 7.4 illustrates a part of synthetic dataset named *ds-1* that includes global, local, collective anomalies and two change points.

7.6.3 Results

7.6.3.1 Experiments on Real Errors

We evaluate the anomaly and change point detection quality of our proposal over the real datasets provided by Yahoo and an IoT solution company. Since Yahoo does not record the change points, we only perform anomaly detection on such datasets. Table 7.1 reports Precision, Recall, F-measure and the number of query to achieve 80% confidence weight

Dataset	Nb Anomalies	Before Active Learning Anomaly Detection			Active Learning Anomaly Detection			Query	Benefit
		Precision	Recall	F-measure	Precision	Recall	F-measure		
real_1	2	4.3	100	8.3	100	100	100	7.0	-2.5
real_3	15	100	100	100	100	100	100	7.0	0.5
real_4	5	55.6	100	71.4	100	100	100	4.0	0.2
real_5	2	100	100	100	100	100	100	1.0	0.5
real_8	10	58.8	100	74.1	90.9	100	95.2	6.0	0.4
real_12	3	100	33.3	50.0	100	100	100	2.0	0.3
real_21	6	85.7	100	92.3	85.7	100	92.3	3.0	0.5
real_22	63	93.1	42.9	58.7	98.1	81.0	88.7	10.0	0.8
real_23	19	38.8	100	55.9	86.4	100	92.7	8.0	0.6
real_24	16	35.6	100	52.5	100	100	100	9.0	0.4
real_25	43	100	14.0	24.5	100	97.7	98.8	3.0	0.9
real_42	44	100	36.4	53.3	100	95.5	97.7	8.0	0.8
Average	**19.0**	**72.7**	**77.2**	**61.8**	**96.8**	**97.8**	**97.1**	**6.5**	**0.5**
real_9	8	100	25.0	40.0	100	25.0	40.0	1.0	0.7
real_11	19	100	26.3	41.7	33.3	57.9	42.3	4.0	0.5
real_30	9	23.1	33.3	27.3	60.0	33.3	42.9	5.0	-0.3
real_38	9	13.8	44.4	21.1	60.0	33.3	42.9	5.0	-0.3
Overall AVG	**9.5**	**57.9**	**50.0**	**44.4**	**87.1**	**77.4**	**78.8**	**5.0**	**0.3**

Table 7.1 – CABD's results over Yahoo datasets.

over 50 Yahoo's datasets.

We present the notable results in the first part of the table. It is not surprising that active learning significantly increases the anomaly detection quality. Without active learning process, the average precision and recall scores is about 72.7% and 77.2%, respectively. In some datasets, F-score achieves 100% such as *real_3* and *real_6*, meaning that all detected points are totally correct without false. After applying active learning to optimize the probabilistic model, both the precision and recall score converge to high values. The results increase to about 96.8% and 97.8% for precision and recall values, respectively. Moreover, the labeling query is very effective shown through the low benefit score (0.5 on average). This means labeling one candidate could reveal 2 other candidates. In further analysis, we analyze the worse results presented in the second part of Table 7.1. Intensively investigating into these results, we realize that the false negatives usually occurs at the boundaries of abnormal data, especially at the ends of collective anomalies.

From two real IoT datasets shown at the end of Table 7.2, we note that the anomaly detection's recall on average are 100% without labeling requirement, this means all abnormal points are recognized. However, the overall F-scores of anomaly and change point detection only achieve about 53.7% and 33.3%, respectively. Based on active learning, the F-score coverages to perfection at 100% after labeling four candidates. These results prove again that our proposal is capable of effectively detecting both anomalies and change points with limited labeling requirements.

7.6.3.2 Experiments on Synthetic Errors

Next, we evaluate CABD on the synthetic datasets which simulate various anomaly types and change points as real scenarios. Similar to real datasets, the detection qualities (both anomaly and change point detection) are measured in two phases: before and after executing active learning. Table 7.2 presents the experiment results of varying anomaly and change point proportion. From this table we note that:

			CABD before Active Learning						CABD with Active Learning							
Dataset	%AP	%CP	Anomaly			Change Point			Anomaly			Change Point			Query	Benefit
			P	R	F	P	R	F	P	R	F	P	R	F		
ds-1	1	1	9.5	94.7	17.3	100	68.4	81.3	100	94.7	97.3	100	100	100	7	0.91
ds-2	5	1	24.9	88.9	38.9	75.0	15.8	26.1	100	83.8	91.2	94.7	94.7	94.7	17	0.92
ds-3	10	1	52.4	68.3	59.3	100	31.6	48.0	100	63.0	77.3	94.1	84.2	88.9	26	0.93
ds-4	15	1	69.7	65.5	67.5	50.0	21.1	29.6	93.2	64.0	75.9	76.5	68.4	72.2	37	0.92
ds-5	20	1	70.9	61.9	66.1	50.0	10.5	17.4	85.0	59.4	69.9	100	52.6	69.0	32	0.95
ds-6	1	2	4.8	100	9.1	91.7	28.2	43.1	100	89.5	94.4	97.4	97.4	97.4	12	0.90
ds-7	5	2	100	58.8	74.0	92.5	94.9	93.7	98.3	60.8	75.2	94.9	94.9	94.9	16	0.93
ds-8	10	2	21.2	92.4	34.5	0.0	0.0	0.0	100	51.4	67.9	92.1	89.7	90.9	31	0.92
ds-9	15	2	79.7	40.9	54.0	61.5	61.5	61.5	79.0	63.4	70.4	87.5	71.8	78.9	51	0.90
ds-10	20	2	68.6	36.9	48.0	59.5	56.4	57.9	88.1	47.2	61.5	95.5	53.8	68.9	60	0.90
ds-11	1	5	3.5	88.9	6.7	98.6	71.7	83.0	100	66.7	80.0	99.0	99.0	99.0	10	0.95
ds-12	5	5	12.9	81.1	22.3	83.3	15.2	25.6	100	46.3	63.3	93.5	86.9	90.1	32	0.90
ds-13	10	5	26.3	79.6	39.5	81.8	18.2	29.8	85.6	43.5	57.6	95.2	80.8	87.4	43	0.90
ds-14	15	5	39.3	86.4	54.0	80.0	12.1	21.1	70.2	59.4	64.4	95.2	60.6	74.1	82	0.85
ds-15	20	5	45.1	60.5	51.6	95.0	19.2	31.9	85.0	40.1	54.5	94.3	66.7	78.1	44	0.93
ds-16	1	10	2.7	100	5.4	100	12.1	21.5	100	73.7	84.8	100	98.0	99.0	11	0.97
ds-17	5	10	15.9	64.6	25.5	99.0	49.7	66.2	100	47.5	64.4	97.7	86.4	91.7	21	0.96
ds-18	10	10	24.5	95.3	39.0	94.6	17.6	29.7	97.6	42.1	58.8	98.1	76.9	86.2	53	0.91
ds-19	15	10	30.2	68.7	42.0	100	8.5	15.7	91.7	40.0	55.7	96.4	66.3	78.6	40	0.94
ds-20	20	10	35.0	55.5	42.9	85.7	12.1	21.1	88.4	41.1	56.1	96.6	56.8	71.5	73	0.90
ds-21	1	20	1.7	100	3.4	0.0	0.0	0.0	100	57.9	73.3	100	84.0	91.3	16	0.98
ds-22	5	20	11.1	100	20.1	100	15.8	27.3	100	37.4	54.4	100	74.2	85.2	59	0.92
ds-23	10	20	29.4	53.1	37.9	100	39.1	56.2	96.9	32.0	48.1	97.3	64.4	77.5	49	0.93
ds-24	15	20	50.2	41.4	45.4	99.4	40.4	57.4	85.3	35.6	50.3	99.0	50.6	67.0	74	0.91
ds-25	20	20	50.7	39.9	44.7	95.7	22.3	36.2	100	34.5	51.3	98.2	40.1	56.9	72	0.92

Average			**35.2**	**72.9**	**38.0**	**79.7**	**29.7**	**39.3**	**93.8**	**55.0**	**67.9**	**95.7**	**76.0**	**83.6**	**38.72**	**0.92**
iot 1	1	1	65.6	100	79.2	0.0	0.0	0.0	100	100	100	100	100	100	4	0.6
iot 2	0.5	1	16.4	100	28.2	100	50.0	66.7	100	100	100	100	100	100	4	0.8
Average			**41.0**	**100**	**53.7**	**50.0**	**25.0**	**33.3**	**100**	**100**	**100**	**100**	**100**	**100**	**4**	**0.7**

Table 7.2 – CABD's results over synthetic datasets with vary anomaly and change point percentages.

First, CABD with active learning significantly improves the anomaly and change point detection accuracy. The average F-score increases from 38% to 67.9% and from 39.3% to 83.6% for anomaly and change point detection, respectively. Remarkably, the active learning takes more benefits at low anomaly percentage. For example: in dataset with 1% anomaly such as *ds-1*, the F-score increases by about 80% from 17.3% to 97.3% after active learning. Similar results are also found in *ds-6* and *ds-11* datasets.

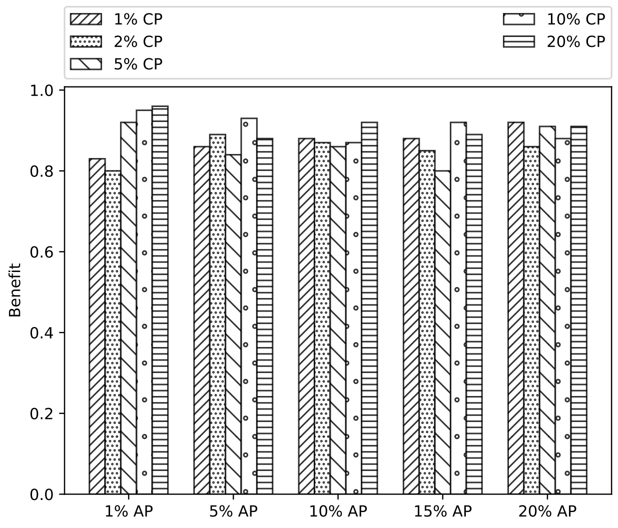


Figure 7.6 – Comparing the query benefit over varying anomaly and change point percentages in datasets.

Second, the query of active learning is highly effective. The benefit score is about 0.88 on average, that means labeling 12 candidates could recognize 100 abnormal points. As shown in Figure 7.6, regardless the changes in the percentage of anomaly and change point, the query benefit is consistent from 0.8 to 0.96. This result demonstrates that the model inputs (the score metrics) calculated from Inverse Nearest Neighbor concept are highly related to detecting anomaly and change points.

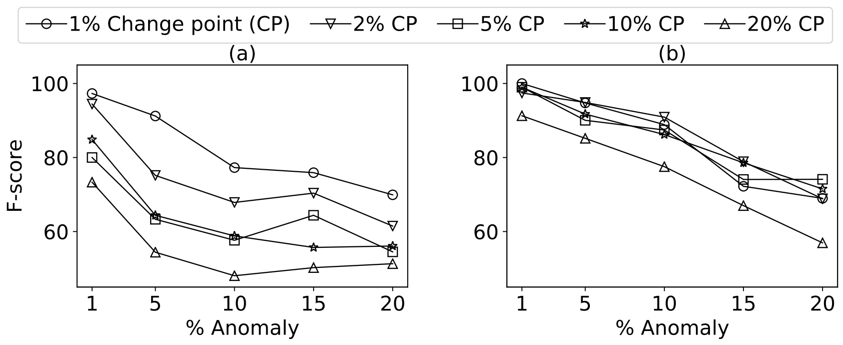


Figure 7.7 – Varying the percentages of anomaly and change points over synthetic datasets. From left to right, the two plots show: (a) Anomaly detection quality; (b) Change point detection quality.

Lastly, as illustrated in Figure 7.7, high percentage of anomalies and change points decreases the efficiency of CABD. In more detail, increasing the anomaly percentage from 1% to 20% makes the F-score decrease by 27.4% and 31% for anomaly and change point detection, respectively. This can be explained that if a single anomaly point is very close to a change point, its spreading pattern based on inverse nearest neighbor is larger than usual. Thus, the metric score of such point is very likely to a change point. This leads the classification model to conflict when labeling this point as a single anomaly.

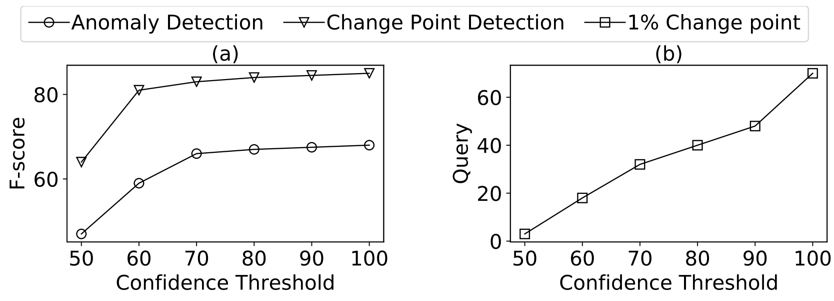


Figure 7.8 – Varying confidence settings: (a) Anomaly and change point detection accuracy; (b) The number of query.

7.6.3.3 Runtime

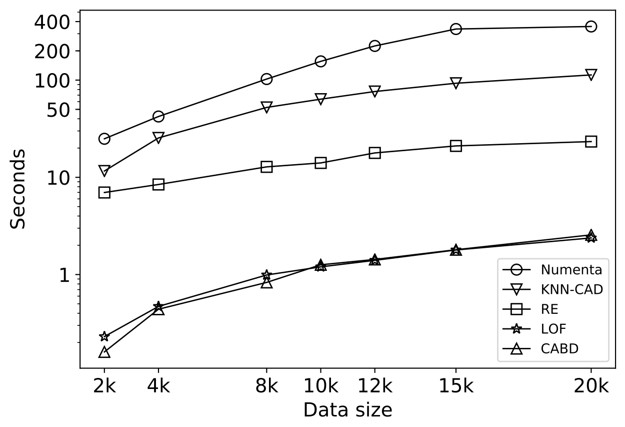


Figure 7.9 – Evaluating the runtime of common anomaly detection algorithms over different data sizes.

We experimentally compare the runtime of CABD with reviewed algorithms on various data sizes from 2000 to 20000. All evaluations are performed on a computer with following configuration: Intel i5-6200U CPU @ 2.30GHz, 2 Core(s), 4 Logical Processor(s), 8GB of RAM and the operating system is 64-bit Windows 10. Since CABD has the active learning steps, labeling time of the end-user is not included. As shown in Figure 7.9, the runtime of CABD roughly equals that of LOF and it is extremely tiny comparing with Numenta or KNN-CAD at all data sizes. More details, with 2000 data points, Numenta and KNN-CAD process in 24.91 and 11.61 seconds, respectively, while the runtime of CABD is only 0.16 seconds. The similar results are also found in the larger datasets. Numenta and KNN-CAD need 356.03 and 113.02 seconds to detect the anomalies in 20000 data points whereas CABD only needs 2.56 seconds. In summary, CABD provides significant better in both detection quality and running time over the state-of-the-art algorithms.

7.6.4 Effectiveness of Active Learning

Dataset	**No AP**	**No CP**	**W/O AL**		**W/ AL**		**Total query**
			AP F-score	**CP F-score**	**AP F-score**	**CP F-score**	
Synthetic	250	190	38.0	39.3	67.9	83.6	38.7
Yahoo	12	-	44.4	-	78.8	-	5.0
IoT	12	10	53.7	33.3	100.0	100.0	4.0

Table 7.3 – Evaluating anomaly detection (AD) and change point detection (CP) qualities on Synthetic, Yahoo and IoT datasets.

In our proposal, the active learning process is an important step to achieve non-parametric algorithm and high accuracy. Summarizing evaluation results from Table 7.3, the detection quality of CABD with active learning always outperforms one of non-active learning. For example, the average f-score of anomaly detection on Yahoo datasets increases by 34.4% from 44.4% to 78.8%. This stems from the fact that active learning could optimize the probabilistic classification model to be more accurate.

Table 7.4 presents the *accuracy score* (acc), *minimum confidence* (cof) of the model in each round. From this table, we note that the model may identify all abnormal points (reach 100% accuracy score) after a limited number of queries. For example, after 4 queries, the model in *real*_1 dataset reaches 100% accuracy. In some round, the model accuracy decreases after labeling a candidate point. This can be described to the following: in case the abnormal point appears very close a change point pattern, the metric score of this point is similar with change point such as high magnitude score, correlation score, and low variance score. Thus, CABD incorrectly detects the point as a change point. Consequently, labeling this point as abnormal conflict with current model awareness. Thereby, it decreases model accuracy.

Round	**real_1**		**real_23**		**real_42**		**real_iot_1**		**real_iot_2**	
	acc	conf	acc	conf	acc	conf	acc	conf	acc	conf
1	0.1	0.5	0.0	0.5	0.0	0.7	0.8	0.7	0.9	0.7
2	0.1	0.4	0.6	0.5	0.1	0.5	1.0	0.6	1.0	0.7
3	0.6	0.4	0.5	0.5	0.2	0.5	1.0	0.7	1.0	0.7
4	0.6	0.4	0.7	0.7	0.6	0.5	1.0	0.8	1.0	1.0
5	1.0	0.6	0.8	0.6	0.4	0.4				
6	1.0	0.6	0.8	0.5	1.0	0.6				
7	1.0	0.8	1.0	0.7	1.0	0.4				
8			1.0	0.8	1.0	0.9				

Table 7.4 – The active learning analysis.

7.6.5 Effectiveness of INN

Similarly active learning, Inverse Nearest Neighbor is a novel concept accelerating anomaly and change point detection accuracy. Moreover, searching INN is a non-parametric algorithm. This makes INN robust to parameter configurations that is one of common limitations in reviewed algorithms. To demonstrate the efficacy of INN in comparison with KNN, we replace INN by KNN in our evaluation. The appropriate K parameter is determined by bruce-foced searching in range from 0 to data size. Such replacement is evaluated on both real and synthetic datasets.

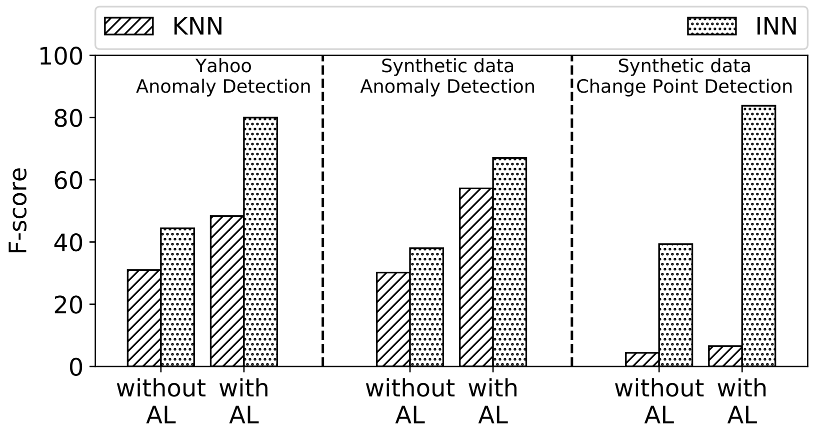


Figure 7.10 – Comparing the effectiveness of INN and KNN in two cases: before and after performing active learning. From left to right, the two plots show: (a) Anomaly detection quality over Yahoo datasets; (b) Anomaly and change point detection quality over Synthetic datasets.

As shown in Figure 7.10, CABD using INN (CABD-INN) shows better performance in comparison with one using KNN (CABD-KNN) in both cases (with and without active learning). Detecting the anomaly in Yahoo dataset, the f-scores of CABD-KNN are reported about 31% and 48.32% for before and after active learning, respectively. Replacing KNN by INN, the f-score significantly increases by 30.48% to 78.8% in case performing AL and by 13.4% to 44.4% before performing AL.

The same results are found in the evaluation over synthetic datasets. CABD-INN outperforms CABD-KNN in all cases (anomaly and change point detection). Especially in change point detection, the f-score of CABD-INN achieves 39.3% before AL and 83.8% after AL respectively while that of CABD-KNN are only about 4.38% and 6.55%. These results prove again the superiority of INN over KNN.

7.6.6 Enhancing Repairing Quality

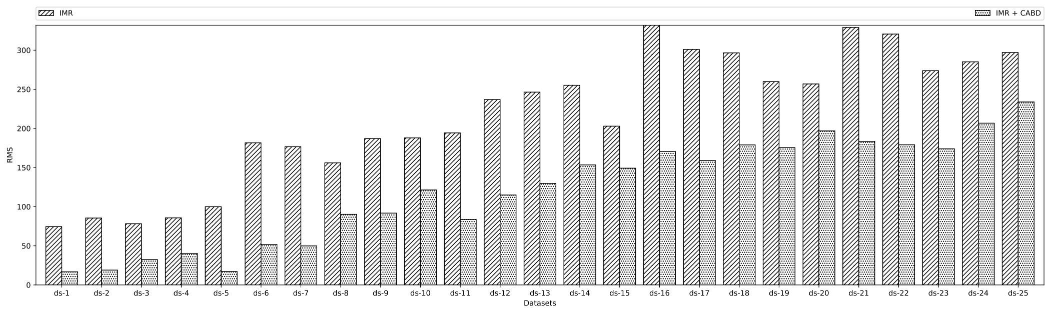


Figure 7.11 – An optimization of IMR.

To minimize the impact of anomalies on data reliability, a repairing process is usually triggered after detecting anomaly. We evaluate the integration of our proposal with a recently proposed data-repairing algorithm, namely *Iterative Minimum Repairing* (IMR) [159, 160]. We aim to show that the quality of the automatic data repairs of IMR could be improved by labeling anomalies using the Active Learning mechanism of CABD. The repairing quality is presented by Root Mean Square (RMS) error that evaluates the distance between ground truth and repaired values. Lower RMS error values indicate better results. Figure 7.11 illustrates the experiment over synthetic datasets with varying anomaly and change point percentages. IMR with CABD for the labelling of the data shows significantly better repairing results than the original IMR (based on random value selections) in all datasets. For example, in dataset *ds-1* and *ds-2*, CABD reduces RMS error 4 times from about 74.5 and 85.4 to 16.7 and 18.9, respectively. Moreover, referring to Table 7.2, with 2000 data points, the average number of labelled data points for synthetic datasets is 38.72. This means that CABD labels about 2% data points to achieve such results comparing with 20% of original IMR. These results prove the utility of our proposal in both improving the repairing quality and in reducing the data labeling effort.

7.6.7 Comparison of Quality

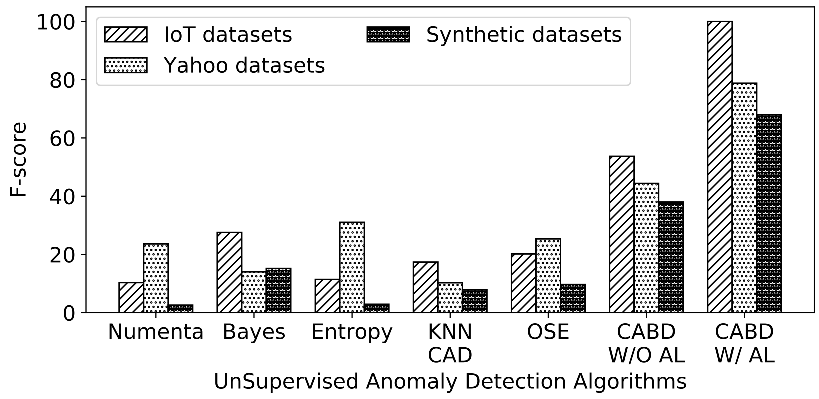


Figure 7.12 – Comparing detection quality with unsupervised anomaly detection algorithms over all datasets.

We do state-of-the-art a number of anomaly detection method but evaluating all of them is extremely heavy. The algorithms are evaluated including Numenta [247], KNN-CAD [246], ContextOSE [276], Multinomial Relative Entropy [142], Bayesian Online detection[277]. The source code and parameter settings for all of the above algorithms are fetched from the *Numenta Anomaly Benchmark* repository [247].

A comparative analysis of detection quality on all datasets (Yahoo, IoT and Synthetic datasets) reported in Figure 7.12 shown that the detection quality presented by F-score of all reviewed algorithms is fairly low when dealing with various anomaly types. The average of F-measure is under 20% even with the recent algorithms such as Numenta or KNN-CAD. That is, they may not deal with a large number of consecutive errors as well as the present of change points. Contrastingly, CABD always shows significantly better results on all cases (with and without applying active learning). The average F-measure scores before and after active learning on all datasets are about 45.3% and 82% respectively.

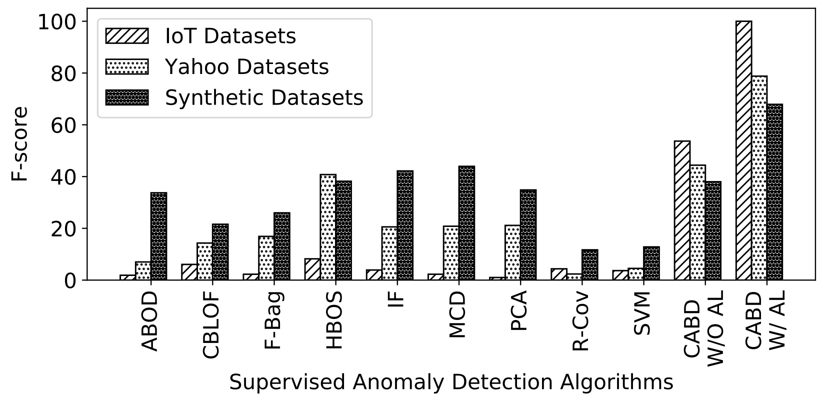


Figure 7.13 – Comparing detection quality with supervised anomaly detection algorithms over all datasets.

In a further study, we compare detection quality between CABD and the most common supervised outlier detection algorithms such as Angle-Based Outlier Detection (ABOD) [278], Clustering-Based Local Outlier Factor (CBLOF) [141], Feature Bagging (F-Bag) [279], Isolation Forest (IF) [264], Minimum Covariance Determinant (MCD) [280], Principal Component Analysis (PCA) [281], Robust Covariance (R-CoV) [266], One-Class Support Vector Machines (SVM) [265]. These algorithms are trained by clean datasets which are eliminated abnormalities. In this evaluation, the source code of such algorithms are obtained from "*Python toolkit for detecting outlying objects*" [282]. Figure 7.13 presents the results on the Yahoo, IoT and synthetic datasets. Again, such results are similar to those in Figure 7.12. That is, CABD always shows significantly better performance comparing with others. For example, on Yahoo datasets, the average F-score of CABD after active learning is 80% while the best of others is only 40% (HBOS algorithm). Similarly, such scores on synthetic datasets are reported about 68% and 38% for CABD and best of supervised algorithms (Isolated Forest (IF) algorithm) respectively.

7.7 Related Work

In this section, we briefly review the concept of nearest neighbor analysis which has been used in the several anomaly detection algorithms. Such techniques detect the abnormality by examining the distance or similarity between two data instances. Normal data instances occur in dense neighborhoods, while anomalies occur far from their closet neighbors [81]. The distance (or similarity) can be calculated in different way. For example: *Euclidean distance* is the wide usage in continuous attributes in [125, 126, 127]. For categorical attributes, a simple matching coefficient is often used [128, 129]. There are two main approaches in Nearest neighbor-based anomaly detection techniques: (1) Using the distance to the k nearest neighbor as anomaly score; (2) Computing the anomaly score from the relative density of each data point.

The original idea of nearest neighbor anomaly detection defines the anomaly score of a data instance as its distance to its k^{th} nearest neighbor which is first mentioned in [130]. This work also has been applied to detect shorted turns in wind turbin-generators in [131]. The basic technique is enhanced by researchers in various aspects. For example: The author in [132, 133, 134] calculates anomaly score from the sum of the distance to k nearest neighbors. [135] counts the number of nearest neighbor within d distance as anomaly score. [136]

designs a simple sampling technique to reduces the complexity of algorithm. A Resolution Outlier Factor (ROF) was proposed in [137]. According to this method, points are outliers depended on the resolution of applied distance thresholds. Conformalized density- and distance-based anomaly detection [246] measure the dissimilarity between observation by combining feature extraction method and conformal paradigm. This was shown to provide effective results for outlier analysis.

The density-based anomaly detection is based on the idea: "An instance that lies in a neighborhood with low density is declared to be anomalous while an instance that lies in a dense neighborhood is declared to be normal". The most well-known algorithm in such technique is Local Outlier Factor (LOF) [125]. For given data instance, the anomaly score is basically the ratio of the average local densities of its k-nearest neighbors over the local density itself. However, LOF ineffectively detects the regions which are not clearly separated. Several researches are subsequently proposed to extend the concept of LOF. The authors in [138] uses symmetric nearest neighbor relationship to define the outlier score. [139] introduces a new variation of LOF named Connectivity-based Outlier Factor (COF)[126] which can detect the abnormality distributed on arbitrarily shaped clusters. The LOF is also combined with other techniques. For example, the works in [140, 141] calculate the anomaly score, named Cluster-Based Local Outlier Factor (CBLOF), from local distances to nearby clusters. The other method is LOCI, a truly density-based method, using the number of circular neighbors around data point as outlier score. Most of existing nearest neighbor anomaly detection algorithms are sensitive to define the k-nearest neighbors. [283] propose a new method using Instability Factor (INS) aiming to overcome the weakness relating the vulnerability of parameter configuration. The enhanced version of INS using "Natural Neighbor" concept is introduced in [284]. A more recent proposal is presented in [142] using relative entropy as the distance measurement.

Additional algorithms for anomaly detection target on time serial data introduce in [285, 286]. Twitter proposes an open-source method named Seasonal Hybrid ESD [269] using seasonal decomposition and statistical metrics to correctly detect anomalies. Skyline [244] and Loda [245] use an ensemble of various weak detections or statistical techniques to enhance the detection performance. The other method named KNN-CAD [246] uses a probabilistic model to interpret the distance based on the *conformal paradigm*. We include evaluation of these methods in our result section.

The key advantages of nearest neighbor-based method are unsupervised and independent with data distribution. This means they are purely data-driven [81]. However, this method fall in two aspect: (1) It cannot handle a sequence of continuous outliers or large collective anomalies due to be sensitive in parameter configurations; (2) It incorrectly detects the seasonal event as abnormality due to lacking frequency checking. In contrast, our CABD approach, using new concept named invert nearest neighbor and applying active learning, could not only detect both single and collective anomalies with better accuracy but also highlight the valuable events.

7.8 Conclusion

In a nutshell, we propose an anomaly and change point detection algorithm. The proposed method combines the novel nearest neighbor concept and active learning to maximize the detection quality while minimizing user interventions. Experimental results in both real and synthetic datasets clearly indicate the superiority of our algorithm over reviewed methods. To further prove the effectiveness of invert nearest neighbor and active learning, we will apply them in not only anomaly detection but also data repairing algorithms for future studies.

8 An Energy Efficient Sampling Algorithm

Energy conservation techniques are crucial to achieve high reliability in Internet of Things services, especially in Massive IoT scenario which stringently requires cost-effective and low-energy consumption for IoT devices. Most of the proposed techniques generally assume that data acquisition and processing consume significantly lower than that of communication. Unfortunately, this assumption is incorrect in IoT scenario, where sensing actions may consume even more energy than the transmission. To deal with these issues, we propose an adaptive sampling algorithm that estimates the optimal sampling frequencies in real-time for IoT devices based on the changes of collected data. Given user's saving desire, our algorithm could minimize the energy consumption of the sensors while ensuring the precision of collected information. Practical experiments have shown the proposed algorithm can reduce the number of acquired samples up to 4 times in comparison with a traditional fixed-rate approach at extremely low error around 4.37%.

8.1 Introduction

The Internet of Things typically consists of a large number of constrained devices widely deployed in the environment [287]. These devices aim to sense the environment and exchange the collected data with collection points or IoT applications. To perform this task, an IoT device has four main components: (1) a sensing subsystem to sense and collect information from the environment; (2) a processing subsystem to manage all device operations; (3) a communication subsystem to transmit the collected data; (4) a power source to supply the energy needed for all device operations. Recently, cloud-based IoT paradigm requires frequent data transmission from connected devices [24]. Such operations quickly drain the device power source capacity, to the point that it may leads to suddenly interrupt all running operations and strongly impact the whole IoT system. In addition, the battery is limited energy capacity. Recharging or replacing such battery is extremely costly or even impossible, because the IoT devices may be deployed in hostile environments (e.g., under the sewer networks or in the deep forest).

In the IoT context, IoT devices must have a sufficient lifetime to fulfill the application requirements. Many approaches have been proposed to minimize energy consumption such as compressing data [164, 288], aggregating data before sending [289, 290], and predictive monitoring [291]. These approaches target on minimizing the radio activity as they assume that the communication subsystem is the most consumed energy source and the energy consumption of sensing and processing subsystems are negligible. However, in IoT devices, the sensing subsystem may consume more energy than the other device components. This is caused by various factors [163]:

- Power hungry transducers: Many sensor types use high power resources to perform sensing tasks such as multimedia sensors or chemical sensors.
- Power hungry Analog/Digital converter: Some sensors need conversing collected data from analog to digital formats.
- Long acquisition time: Some sensing operations may require from seconds to minutes.

Therefore, the consumed energy of the sensing subsystem need to be concerned. Reducing the energy for communication subsystem may be not enough. The energy-efficient system need to decrease the number of acquisitions (collecting information by using sensing subsystem).

Typically, the energy consumption of all subsystems (including sensing, processing and communicating) highly relates to sampling frequency either directly or indirectly. Thus, effectively reducing energy consumption could be achieved by decreasing the sampling frequency. In practice, the data sampling rate should depend on the changes in the collected data. This means that if the variation of collected data is high, the sampling frequency should be increase to collect sufficient information for further analysis. Based on such idea, we propose an *Online Adaptive Sampling Algorithm* (OASA) that estimates in real-time the optimal sampling frequency for sensors based on historical data. Given user's saving desire, our algorithm minimizes the energy consumption of the sensors according to this saving desire while maintaining the accuracy of collected information. Practical experiments have shown that the proposed algorithm reduces the number of acquired samples up to four times in comparison with a traditional fixed-rate approach at extremely low error around 4.37%. The superiorities of our proposal are summarized below:

- OASA estimates the optimal sampling frequency online.
- OASA is more general than other solutions because it does not require any assumption.
- OASA allows the end-user to manage the saving energy level. This increases the flexibility of our algorithm.
- The performance of OASA is practically demonstrated to be remarkably higher than existing solutions.

8.2 Related Work

Energy efficiency for constraint devices is a highly interesting topic in the context of Internet of Things. For instance, in the agriculture scenario where IoT devices are widely distributed in a large region, optimizing the energy consumption may extend the *device life circle* as well as significantly reducing the maintenance cost.

Considering the fact that waking-up, collecting, and pre-processing operations consume a similar proportion of energy comparing with transmitting, the fundamental idea of adaptive sampling technique is to adapt the sampling rate to the changes of observation based on specific criteria while ensuring the precision of outcome information. In this section, we catalog the adaptive sampling approaches based on such criteria.

Send-on-delta sampling: is the most commonly used in wireless networks. The original of such approach is the level-crossing sampling at late 1950s based on the idea "the most suitable sampling is by transmission of only significant data, as the new value obtained when the signal is changed by a given increment" [292]. Due to its popularity, there are various temps expressing this strategy such as event-based sampling [293], magnitude-driven sampling [294] or deadbands [295]. Formally, given threshold δ, a message has value y_i at t_i is sent if and only if

$$(y_i - y_k) > \delta \tag{8.1}$$

With y_k is the last message sent at t_k. To prevent babbling-idiot failure on such approach, the min and max sending time, denoted by T_L and T_H, respectively, are defined as the boundary of sampling interval ($T_L \leq t_i - t_l \leq T_H$).

Integral Sampling uses the concept of integral or energy of the error to deal with small oscillations in the signal. The message is sent if the accumulated error of sampling, denoted by CES, is greater than a pre-defined threshold ξ. The min and max-send-time are also applied. The CES value of a signal $x(t)$ is the difference between $x(t)$ and accumulated value from the most recent sample $x(t_k)$.

$$CES_{x_t} = \int\limits_{t_i}^{t_{i-1}} [x(t) - x(t_{t-1})]^2 dt \tag{8.2}$$

where $i = 1, 2, ..., n$ is the number of sample taken from t_0 to t_n [296].

Predictor-based sampling uses a model to predict the next measure based on past values. The message $x(t)$ is sent if it significantly differs with the predicted value $\hat{x}(t)$. The criterion of the difference may reuse either sen-on-data or interval sampling. The model is built from a simplified statistic using linear extrapolation [297]. To maintain the high information quality, the predictor is used in the receivers to extrapolate the signal value until receiving the new message. However, updating receiver predictor requires at least two samples. This reduces the efficiency of such approach.

Gradient-based integral sampling is an extension of integral sampling approach with the optimization of wake-up energy consumption. This method based on the fact that waking-up device consumes considerably larger than collecting message. Hence, the next wake-up time is automatically adjusted to the current gradient of the signal [298]. To the avoid the worst scenario which the signal gradient is zero, a max-sleep-time is defined.

Sigmoid-based sampling uses a sigmoid function to estimate the changes of sampling rate based on the variance of the last windowed signal [299] [26]. Let denote the last message be $x(t)$ belonging a signal window size W, the variance is the absolute difference of between $x(t)$ and $x(t-1)$ over the average value of W. Next, such variance is compared with a predetermined threshold before calculating the new sampling rate is the multiplication of current race and the sigmoid function of such variance. Such new rate is limited from 0 to 2 as a result of sigmoid function properties.

All reviewed algorithms are designed based on specific assumptions. For example, Send-on-delta sampling assumes that "the most suitable sampling is by transmission of only significant data", Gradient-based integral sampling bases on "waking-up device consumes considerably larger than collecting message". This leads to reduce the flexibility of these approaches. In addition, there are no methods for handling the user's interests related to the power-saving degree.

8.3 Adaptive Sampling Algorithm

To mitigate discussed limitations, we propose a light-weight adaptive sampling algorithm to improve the energy-efficiency while maintaining the accuracy of collected information. In this section, we first present the overall algorithm. Then, we briefly explain each step along with related definitions.

8.3.1 Preliminaries

Definition 1(Absolute first difference): The absolute value of first difference of $\theta_i(p)$, denoted as $\triangle\theta_i(p)$, which is defined that:

$$\triangle\theta_i(p) = |\theta_i(p) - \theta_{i-1}(p)|,\ i = 1, 2, 3, ..., n \tag{8.3}$$

Definition 2(Absolute Second difference): The absolute value of second difference of $\theta_i(p)$, denoted as $\triangle"\theta_i(p)$, which is defined that:

$$\triangle"\theta_i(p) = |\triangle\theta_i(p) - \triangle\theta_{i-1}(p)|,\ i = 1, 2, 3, ..., n \tag{8.4}$$

Definition 3(Sigmoid function): Sigmoid function refers to the special case of the logistic function having a characteristic S-shaped curve, which is defined that:

$$S(x) = \frac{1}{1 + e^{-x}} \tag{8.5}$$

8.3.2 Algorithm Overview

The general idea of our proposed algorithm is to dynamically adapt the sampling frequency to the changes of sensed data. Obviously, a higher frequency is preferred to aware the context where there are significant changes (high variance) in the sensed data. For example, in forest fire warning services, the sudden increases of sensed temperature are considered as notable events (e.g., forest fire). Increasing the device frequency to collect more data may help to deeper investigating such events. In contrast, if the observed values are hardly fluctuated, decreasing the frequency is desired to save energy in data sampling, processing, and transmitting.

To actualize the idea, a enhanced sigmoid function is exploited to quickly adapt the sampling frequency, described as:

$$f_{change} = n + \frac{1 - n}{1 + e^{-n*D}} \tag{8.6}$$

with:

$$D = \frac{\triangle\theta_i(X_i) - \frac{n+1}{2} * (\frac{1}{N}\sum_{j=i-N}^{i} \triangle\theta_i(X_j))}{\frac{1}{N}\sum_{j=i-N}^{i} \triangle\theta_i(X_j)} \tag{8.7}$$

In the equation 8.6, n is the user desire of power saving and D presents the sudden changes of coming data comparing with sliding windows based on last N data points. It is reasonable to compare the change of the absolute difference between X_i's absolute first difference over the mean value of such difference of last N data points. If D is sufficiently large, this means the current change is overwhelming in comparison with recent history. Then the sampling frequency is adapted with this change. In contrast, if the value of D is negative, which means the values change is not large enough, the sampling frequency is reduced to save the energy.

Our proposed algorithm exploits the extended sigmoid function corresponding with a natural sampling process which is robust with faulty in sensed data. This scheme ensures that the new frequency is calculated not only based on lasted sensed data but also historical data. In more detail, a anomaly value in sensor reading, which may be significantly higher than the average change recently, does not strongly impact on the next sampling frequency. The frequency significantly changes when there are consecutive changes in sensed data. As the result, our approach effectively determines whether the device energy is either consumed or conserved based on the trend of the sensed data rather than uncertain changes on last value. The pseudo code for implementing our proposal is presented as below:

```
Algorithm 1: Adaptive Frequency for of newest data point $X_i$
Input: $X_i$, window size $N = 50$
Output: New frequency $f_{new}$
1. Initializing: Window W = $\{X_{i-N}, X_{i-N+1}, ...., X_i\}$
2. Calculating changing degree:
      $D = \frac{\triangle\theta_i(X_i) - \frac{n+1}{2} * (\frac{1}{N}\sum_{j=i-N}^{i}\triangle\theta_i(X_j))}{\frac{1}{N}\sum_{j=i-N}^{i}\triangle\theta_i(X_j)}$
3. Calculating the new frequency:
      $f_{new} = n + \frac{1-n}{1+e^{-n*D}}$
return $f_{new}$
```

8.4 Experimental Evaluation

8.4.1 Metrics of Measurement

For evaluating the efficiency of the proposed algorithm, we use two metrics:

- *Normalized Mean Error* (NME) indicates the overall goodness of fit after normalizing between the original signal and reconstructed signal from sampled data. This factor is defined as:

$$NME = \frac{1}{n}\sum_{i=1}^{n}|\hat{x}_i - x_i| * 100\% \tag{8.8}$$

  with $\hat{x}_i$ denotes the normalized ith data in the reconstructed signal, x_i represents the normalized ith data in the original signal and n is the size of signal.

- *Resource Saving Factor* (RSF) indicates the conserved resources based on the reduction in transmitted messages be defined as

$$RSF = \frac{\hat{m}}{n} \tag{8.9}$$

  with $\hat{m}$ and n are the size of sampled data and original data, respectively.

8.4.2 Benchmark Datasets

National Oceanic and Atmospheric Administration (NOAA) datasets: The NOAA provides a set of real-time data about water-quality from a place named "Jamestown." In order to reasonably compare with state-of-the-art approaches, we choose the same dataset and monitoring duration ranges with them which are turbidity and DO from 15 December 2016 to 15 March 2017.

IoT datasets: The data is collected from 2 real CO2 sensors deployed in the working space with the sampling interval of 1h for a sample.

8.4.3 Evaluative Simulation

To assess our proposal performance, we simulate the evaluation closed to real deployments on devices. First, the first sliding window (first N data points) is obtained under original sampling frequency. Then the next samplings are calculated by OASA algorithm. We derive the data values for these samplings from original dataset by using a *linear interpolation method* [300]. The process is repeated until the next sampling exceeded original dataset. The obtained dataset by our method is up-sampled to the size with original dataset and normalized before calculating the metric of measurement (NME and RSF). The process is presented in the algorithm 2.

```
Algorithm 2: Evaluation Process
Input:   Dataset X, Window size N, Saving desire n
Output: $NME, RFS$
1. $f_{curr} \leftarrow f_{const}$
2. Y = [ $x_0, x_i, ...., x_n$ ], i = N, $x_i \in X$
3. While $i < size(X)$ do
       $f_{new} \leftarrow$ MyAL$(Y, N, n)$
       $X_{next} \leftarrow$ Interpolation$(X, f_{new})$
       $f_{curr} \leftarrow f_{new}$
       $Y = Y \cup \{X_{next}\}$
       $i = i + \frac{1}{f_{new}}$
4. $\hat{X} \leftarrow UpSample(Y)$
   $\hat{X}_{norm} \leftarrow Normalize(\hat{X})$
   $X_{norm} \leftarrow Normalize(X)$
Return $NME(X_{norm}, \hat{X}_{norm}), RSF(Y, X)$
```

8.4.4 Results

8.4.4.1 Varying user desire n

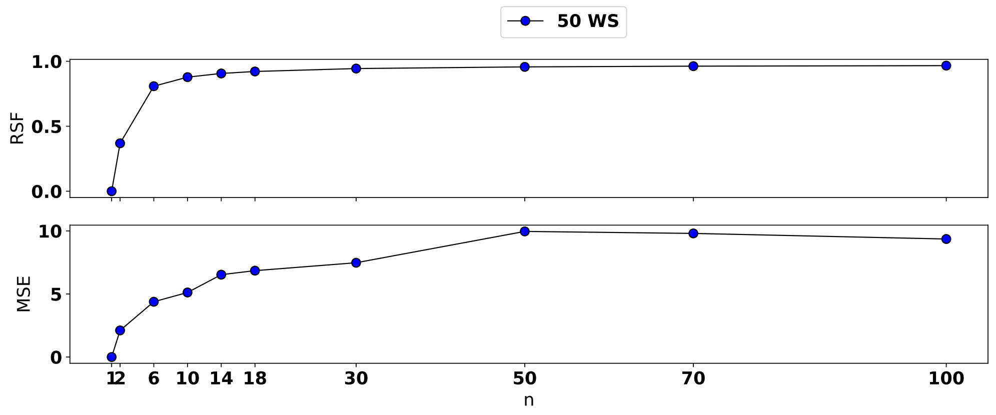


Figure 8.1 – Varying user desires over DO datasets with window size = 50.

Figure 8.1 presents the results on varying user desire (denoted by n), for DO dataset with fix window size (equaling 50). First, as shown in the top plot of Figure 8.1, our proposal shows better energy saving while increasing user desire n. This means that high saving desire leads to less transmitted messages. If the value of n equals 1, the RSF value equals 0 due to no saving resource. Remarkably, the RSF value is significant increase at the low user desire (from 0 to 18) and slightly stable at the higher values (from 18 to 100). This is mainly caused by the fact that the n value is the upper asymptote of the frequency and unaffected the algorithm output. At a sufficient large n, the RSF value is stable.

It is not surprising that the changes of MSE and RFS is similar while increasing user desire value. This is illustrated in the botton plot of Figure 8.1 wherein the MSE value is significantly increase while increasing n value from 1 to 6 then it slowly coverage to stable value since the MSE is proportional to RFS. This could be explained that higher RFS leads to less sampled and transmitted data. Therefore, the reconstructed data is more different with original data. As a result, the value of MSE is increase. The same behaviours are also found in IoT datasets and presented in Figure 8.2.

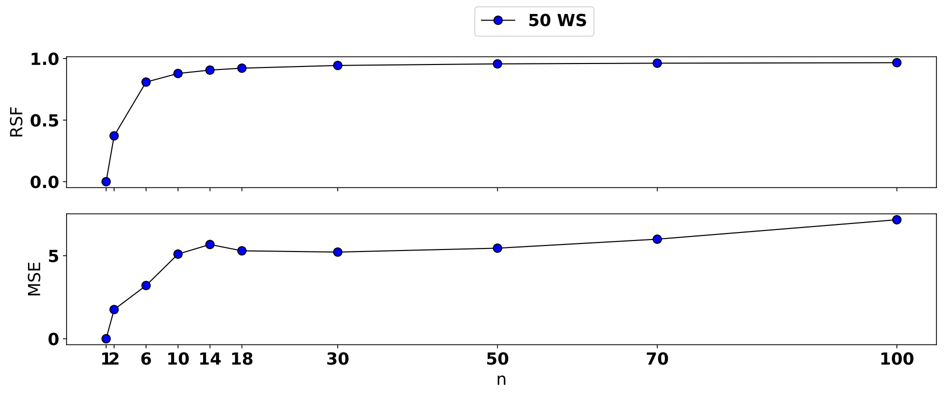


Figure 8.2 – Varying user desires over IoT datasets with window size = 50.

In summary, as presented in Figure 8.1 and 8.2 over the DO and IoT datasets, our proposal is not only capable of preserving the device's energy based on user desires but also robust with the excessively large values of user's saving desire .

8.4.4.2 Varying window size

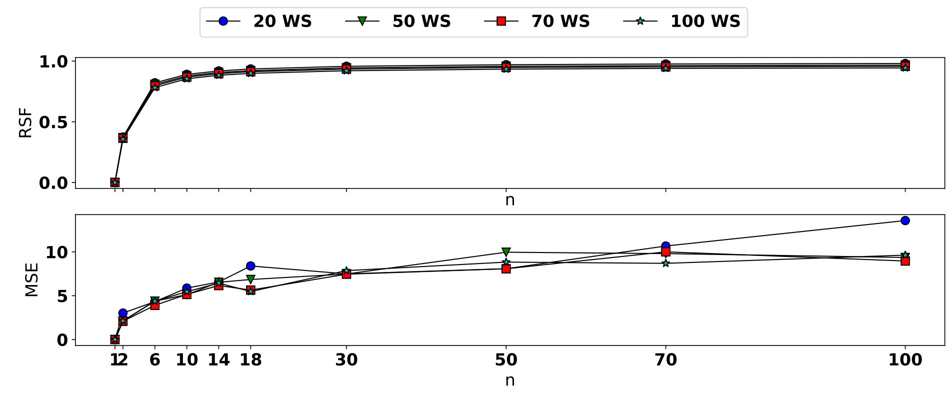


Figure 8.3 – Varying user desires and window sizes over DO datasets.

Figure 8.3 reports the result by varying the window size (ws) for different n values on DO dataset. As illustrated in this figure, we note that both RSF and MSE values are independent with ws configurations. While increasing the window size from 20 to 100, the RSF is stable around a constant value. This is resulted from the superiority of D function. In more detail, the D function compares the current changes degree with the mean of first derivative of historical data. By increasing window size to obtain more history does not strongly impact on this mean, that is, the optimized frequency is well-balanced with the increase of window sizes. The similar result is found in IoT dataset presented in Figure 8.4. This result again consolidates the effectiveness and consistency of our algorithm.

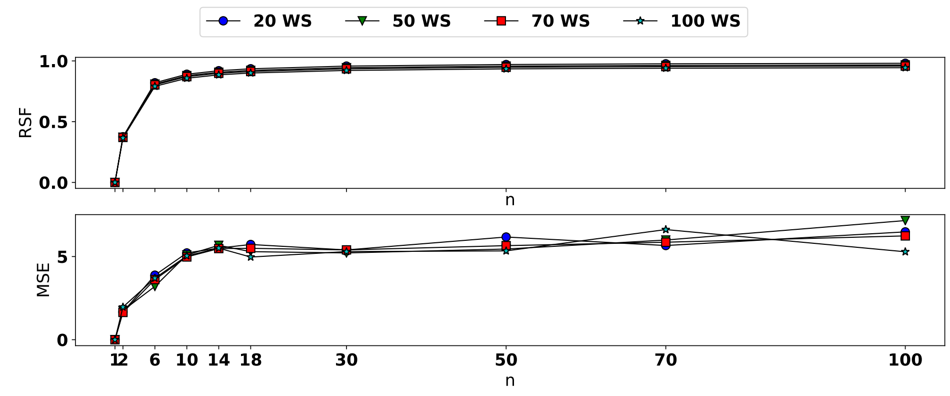


Figure 8.4 – Varying user desires and window sizes over IoT datasets.

8.4.5 Comparison of Results

We do state-of-the-art some adapting sampling algorithms for saving energy but evaluating all of them is extremely heavy. Hence, we compare our algorithm with DDASA [299], ASA [26] which are the most related to our approach. For transparency, the competitor results are derived from original paper.

In Table 8.1, we compare the NME of our algorithm with that of competitors in the same sample rate. The better algorithm has lower NME. As illustrated in Table 8.1, our proposal shows superiority over the competitors. To reduce original dataset to 297 and 421 samples, the NME values of our approach are about 4.96% and 4.07% in comparison with 9.99% and 8.43 % of DDASA, respectively. This means that the reconstructed data from our selected points is more likely to original data than others. Moreover, our approach is more consistent and robust than the competitor in high power saving scenarios. This is demonstrated by the minor change of NME when significantly reducing the number of samples. In more detail, our NME variance is around 0.79 compared with 13.6 of DDASA when decreasing the sample size from 1064 to 297.

In summary, comparing with other approaches, our algorithm has better performance demonstrated by lower and more consistent in MSE values.

	DDASA	DDASA	DDASA	ASA	DDASA	Fixed Rate
	(t=0.03)	(t=0.02)	(t=0.015)		(t=0.01)	Sampling
Number of Samples	297	421	548	637	1064	2182
Competitor's NME	9.99 %	8.43 %	5.31 %	5.52 %	1.62 %	0
OASA NME	4.96 %	4.07 %	4.37 %	3.86 %	2.84 %	0

Table 8.1 – OASA in comparison to the competitors.

8.5 Conclusion

In this work, we introduce an online adaptive sampling algorithm to estimate the optimal sampling frequency base on the changes in historical data. In addition, the proposed algorithm allows the end-users to control the saving energy levels. Hence, it has been demonstrated to reduce energy consumption in various use-cases while ensuring the accuracy of collected data. Our practical experiments have shown the proposed algorithm can reduce the number of acquired samples up to 4 times in comparison with a traditional fixed-rate approach at extremely low error around 4.37%. Consequently, the devices could significantly reduce the energy consumption in IoT devices.

9
Conclusions and Outlook

9.1 Conclusion

We have been witnessing the explosion of Internet of Things and its positive impacts on several real-life aspects. Based on the fact that IoT consists of small things, widely distributed, and constrained capabilities in term of storage and computation capabilities. In addition, the raw data collected from IoT Things is massive, heterogeneous, and contains a vast amount of redundant information. For these reasons, several issues mainly related to interoperability and reliability in Internet of Things have arisen.

To mitigate these issues, this thesis proposed the solutions to enhance both the interoperability and reliability. Through this dissertation, the following objectives are achieved:

- *Identify the current states and challenges of the cloud-based Internet of Things:* In the scope of this thesis, we highlight the related work and issues related to interoperability, data reliability, and device reliability.
- *Propose an IoT framework to interoperate IoT device connections using connectors:* The proposed solution represents a framework that simplifies establishing the connection process from heterogeneous IoT Things to cloud-based platforms by using connectors. In addition, our platform provides well-supplied web services based on a resource-oriented model. It allows the end-users to easily discover and perform management operations on the connectors. Another innovative aspect of our solution is to facilitate and speed up the data acquisition process from open data sharing web services like Accuweather, OpenSensorIO. The interoperability with other such implementations is preserved by using several ongoing IoT standardization methods.
- *Introduce an IoT framework to maximize usable knowledge from IoT data using virtual sensors:* The proposed framework simplifies creating and configuring virtual sensors with the programmable operators (rule, formula or function). To produce high-level information from collected data, these VSs are linked together to create a topology, namely logical data-flow (LDF). In addition, the LDF outcomes are formed under JSON-LD format, which is used to generate interpretable data across different IoT platforms. In this way, it significantly increases the interoperability of our solution. A web-based virtual sensor editor is implemented on the top of the framework to facilitate the creation and configuration of LDF.
- *Present a semantically Descriptive Language for Group of Things:* We introduce a novel description language semantically presenting the group of Things, namely

Assets, in Massive IoT scenario. A WoT architecture is also presented to fully exploit the benefits of the proposed language. Such combination is not only capable of presenting, accessing, and managing the Asset but also speeding up the IoT application development. The effectiveness of the designed solution is ensured by choosing and combining ongoing technologies and IoT frameworks that have been practically demonstrated in real use-cases.

- *Propose an Active Learning method for errors and events detection in time series*: The proposed algorithm effectively detects both errors and events in a single algorithm powered by active learning while minimize user involvements. This is achieved by introducing a novel concept of nearest neighbor and integrating it into detecting and learning processes. Our experiments show that labeling a very limited number of data points significantly increases the error and event detection quality. In addition, this quality is controlled by end-users through the confidence weight of a classification model, which is then used as the termination condition for the active learning procedure.

- *Propose an Energy-Efficient Sampling Algorithm*: The proposed algorithm minimizes energy consumption by estimating an optimal data collection frequency in real-time based on historical data. Given user's saving desire, our algorithm minimizes the energy consumption of the IoT devices while ensuring the precision of collected data. The practical experiments have shown that the proposed algorithm reduces the number of samples up to four times in comparison to a traditional fixed-rate approach.

All proposed solutions have been implemented and operating in a cloud-based IoT platform of a start-up company. This platform aims to provide cloud-based services using Internet of Thing technologies to solve real-life problems. In order to handle the connection from heterogeneous IoT devices, our solution related to the interoperability at device communication level is deployed at the communication layer of such platform. Then, they use the second solution about virtual sensors to maximize usable knowledge from sensed data. In addition, errors and events detection algorithm is implemented to ensure data quality. On the other hand, the energy efficient sampling algorithm is implemented on their IoT devices to reduce energy consumption, especially on constrained-resources devices.

9.2 Perspectives and Future work

With the desire to bride the gaps in interoperability and reliability in Internet of Things, this thesis proposed various solutions spreading over architecture, models, and algorithms and cover most of the layers of the IoT architecture. However, due to the variety of the topic, several aspects are unanalyzed in details so that there are big rooms for improvement.

- *Organizational Interoperability:* Throughout the dissertation, our solutions only target the syntactical and semantic interoperability. The organizational interoperability (the highest level of interoperability) did not focus. It would be interesting to see how to combine our solutions about syntactical and semantic interoperability to achieve the organizational interoperability.

- *Leveraging active learning for data repairing:* In Chapter 7, we use the active learning to improve the errors and events detection quality. Based on the user interaction, the proposed algorithm effectively identifies and distinguishes between abnormal data caused by errors and one caused by events. The active learning is applied to maximize the detecting performance while minimizing the user interaction. Conceptually, active learning could be used in data repairing in the same purpose. We believe that labeling the "important" points in datasets may significantly increase the repairing quality.

- *Enhancing virtual sensor capability:* More and more complex IoT applications and services have been emerging due to the rapid growth of IoT. However, the virtual sensors presented in Chapter 5 only support mathematical operations. Thus, we need to enhance the functionality of such operations. For example, we could provide customizable operations that allow end-users to develop or configure their desired operations.

- *Comparing the connector framework with existing approaches:* Due to the rapid grow of IoT frameworks, there are several emerging frameworks targeting the interoperability of IoT connectivity. Thus, we will compare their advantages with our proposal. Based on this analysis, we could define a road map for such proposal.

- *Device management in Massive IoT:* Another considering topic is the device management in Massive IoT context. Most of the current device management mechanisms require bidirectional communication. However, in massive IoT scenario, the downlink connection from the network to the device is significantly limited in term of bandwidth and the number of message per day. Therefore, a novel device management mechanism to bride these gaps is a potential research direction.

Appendix A
Résumé de la Thèse en Français

A.1 Introduction

L'internet des objets (IoT), aussi connu comme "Internet de tout" (IoE - Internet of Everything), est un nouveau paradigme qui suscite l'attention dans la sphère Internet. Li'dée de base de l'IoT repose sur l'interconnexion de nombreux "objets" " via un système d'adressage unique, dès lors que les "Objets" sont identifiables, exemple : des capteurs, des actionneurs, des tags connectés ou des smartphones [2][1]. Il n'est par surprenant que cela soit considéré comme une nouvelle révolution d'internet qui impacte la vie quotidienne sur plusieurs aspects, dans la mesure où "US National Intelligence Council" a listé l'IoT parmi les six "technologies civiles disruptives" pouvant impacter l'économie américaine" [7]. En 2011, le nombre d'appareils connectés à internat a dépassé le nombre d'être humains sur la planète. D'ici la fin 2020, 212 milliards d'objets intelligents IoT seront déployés dans le monde et d'ici 2025, les meubles, les documents papier, etc [8][9].

Malgré une croissance rapide et des opportunités illimitées (telles que des solutions de transport intelligentes, des réseaux intelligents et systèmes de contrôle de la pollution de l'air) dans la vie de tous les jours, l'IoT a été toujours confronté à de nombreux défis critiques. La plupart d'entre eux liés à l'intégration, la collecte, le traitement ou le partage des données des objets de l'IoT, émergent lors d'un déploiement à grande échelle [10]. Ces défis sont liés au nombre important d'objets hétérogènes couplés aux contraintes de stockage, de traitement et de communication. Le volume de données brutes collectées provenant des objets est considérable et hétérogène, il contient un nombre important d'informations anormales et redondantes. [2]. Les solutions antérieures qui traitaient des enjeux similaires ne conviennent plus pour deux raisons principales : (1) elles sont trop lourdes pour fonctionner sur des objets de petite taille (2) Elles ne sont pas automatisables, et ne peuvet par être traitées par des humains vu la quantité de donées en jeu. Les fournisseurs de services IoT ont alors recherché des solutions innovantes pour résoudre ces problématiques. En parallèle, le cloud computing est apparu comme une technologie de rupture qui proposaient d'énormes capacités de stockage et de traitement. La mise en oeuvre de ces technologies offraient des opportunités de traiter, même partiellement, les enjeux de l'IoT. Par exemple, les périphériques IoT à fortes contraintes de taille, ne peuvent pas réaliser des traitements de données complexes en local, ce qui pousse à transmettre ces données vers des unités de traitement plus puissantes (comme les passerelles ou les routeurs), avec une limite de scalabilité importante. Dans ce contexte, la puissance de traitement illimitée du cloud computing peut être utilisée pour mener à bien les tâches complexes, pour un nombre important d'appareils tout en maintenant une grande évolutivité. Par ailleurs, l'IoT peut tirer parti de la capacité de stockage infinie du cloud pour stocker des données de manière plus sécurisée et facilement accessible depuis n'importe où. Par conséquent, l'intégration entre l'internet des objets et le cloud est inévitable, pour créer un nouveau Paradigme informatique appelé *CloudIoT* ou *cloud-based IoT* [12][14].

Bien que l'intégration de l'IoT et du cloud computing offre de nombreux avantages, les scénarios complexes de l'IoT dans le cloud posent plusieurs défis qui ont retenu l'attention de la communauté des chercheurs, tels que la sécurité et la confidentialité, le volume important de données, les performances, l'informatique géodistribuée (fog computing), etc. [15]. Deux de ces défis sont ciblés dans nos travaux, à savoir: *l'interoperabilité* et la *fiabilité*.

L'interoperabilité Le défi le plus critique dans l'IoT basé sur le cloud est le manque de standard unique en plusieurs points des dispositifs, allant des plates-formes, aux services et aux applications [16]. En outre, Les plates-formes IoT du cloud ont généralement été conçues comme des solutions verticales isolées pour fins spécifiques [18]. Pour s'intégrer à ces plates-formes, les fournisseurs de solutions IoT doivent analyser en détail les exigences relatives au matériel, aux logiciels et aux sous-systèmes qui sont étroitement spécialisés au contexte d'usage [?]. D'autre part, les nouveaux types d'appareils IoT ainsi que leurs formats de données propres ne sont pas conformes aux normes IoT existantes ou qui apparaissent chaque jour. Ceci conduit à une grande hétérogénéité lors des déploiements de solutions IoT à grande échelle.

La fiabilité De nombreuses applications IoT (telles que la détection de feux de foret, la prévision des tremblements de terre) sont critiques pour la société et nécessitent une grande fiabilité des technologies sous-jacentes. Cela signifie que ces applications doivent fournir des services de haute qualité, même en cas de bruit ou autres defaillances dans les données collectées. La consommation d'énergie des equipement IoT a également un impact important sur la fiabilité globale du système.

- **Fiabilité de la donnée:** En conséquence d'une croissance rapide, un tres grand nombre de données sont collectées depuis des milliards de sources interconnectées. Cependant, ces données ne sont pas toujours fiables en raison de de nombreux facteurs défavorables, telque l'echelle de déploiemente [301], les contrainte d'energie ou de capacité de calcul [22], les pertes de connexion [23]. Malheureusement,Les applications industrielles, telles que la surveillance d'installations industrielles ou la détection de pannes, nécessitent l'intégrité et la fiabilité des données. Des données manquantes ou aberrantes peuvent déclencher de fausses alertes ou déclencher des processus de correction qui peuvent avoir des impactes economiques importants..
- **Fiabilité des devices:** Le paradigme IoT nécessite une transmission fréquente des données des devices qui sont connectés au cloud [24]. De telles opérations épuisent rapidement les capacités de la batterie des devices, au point de provoquer l'arrêt soudain de toutes les opérations en cours, telles que la collecte de données ou la connexion réseau. Par conséquent, les techniques de conservation de l'énergie sont essentielles pour produire des services IdO avec une fiabilité élevé, en particulier dans un scénario (LPWAN) scénario tres exigeant sur le cout du device et sa consommation d'energie [25].

Dans cette thèse, notre objectif est de traiter les problèmes d'interopérabilité et de fiabilité posés par un déploiement à grande échelle. Les contributions clés de cette thèse peuvent être résumées comme suit:

Interoperabilité des solutions IoT

- *Une méthode pour inter-connecter les devices IoT à l'aide de connecteurs:* Cette solution prend la forme d'un framework IoT innovant qui facillite la création de connecteurs "cloud" pour produire des connexions hétérogènes à partir d'objets IoT. En règle générale, le connecteur est un morceau de code spécifique utilisé pour embarquer la connectivité d'un objet dans un simple service Web RESTful. Ainsi, il réduit considérablement les efforts de création et de configuration de la connexion de la plate-forme IoT "cloud" à des objets hétérogènes. De plus, notre proposition peut aider les utilisateurs finaux à récupérer rapidement les données de diverses sources de données via les connecteurs créés à partir de modèles donnés. L'interopérabilité avec d'autres implémentations est préservée en utilisant les standardisations IoT en cours.

- *Introduction d'un framework IoT avec la notion de capteurs virtuels pour augmenter ce que l'on appelle le "usable knowledge":* Notre infrastructure simplifie la création et la configuration de *capteurs virtuels* (CVs) avec des opérateurs programmables tels que des règles, des formules ou des fonctions. Ces VS pourraient être reliés entre eux pour constituer un réseau dénomé *logical data-flow* (LDF) permettant notamment de produire des informations de haut niveau à partir des données collectées Les sorties du LDF sont au format standard JSON-LD, largement utilisé pour générer des données interprétables sur différentes plates-formes IoT. L'interopérabilité de notre solution augmente ainsi considérablement.

- *Présenter un langage sémantiquement descriptif pour un groupe d'objets:* La solution proposée est une nouvelle description sémantique, à savoir *Web of Things – Asset Description* (WoT-AD),qui décrit sémantiquement un groupe d'objets (également appelé *asset*) comme un objet homogène. WoT-AD aide les utilisateurs finaux à découvrir et à accéder aux ressources, entités et services de l'asset. Nous fournissons également un "framework" léger intégré à WoT-AD pour activer WoT dans un scénario de "Massive IoT". Notre proposition consiste non seulement à modéliser efficacement l'asset, mais également à simplifier le développement d'applications de "mash-up" pour des utilisateurs avec des niveaux de competences diverses.

IoT Solutions de Fiabilité

- *Proposez une méthode d'apprentissage actif pour la détection des erreurs et des événements dans les séries chronologiques:* Notre méthode détecte efficacement les erreurs et les événements dans un seul algorithme optimisé par un apprentissage actif. En outre, la qualité de la détection est contrôlée par l'utilisateur final par observation de la fiabilité de la classification, qui est ensuite utilisée comme condition de finalisation du processus d'apprentissage actif.

- *Proposer un algorithme d'échantillonnage économe en énergie:* L'algorithme proposé minimise la consommation d'énergie en estimant en temps réel la fréquence optimale de collecte des données sur la base des données historiques. Notre solution est suffisamment légère pour être déployée sur des objets IoT soumis à des contraintes de puissance de calcul et de stockage.

A.2 Travaux connexes et défis

A.2.1 Interopérabilité dans l'IoT

Pour accroître l'interopérabilité dans l'IoT, les chercheurs ont mis à profit plusieurs approches et technologies issues d'autres domaines, tels que le Web sémantique, cloud computing et le fog computing. Dans cette section, nous présentons un aperçu des approches actuelles ainsi que des défis à relever pour parvenir à l'interopérabilité.

A.2.1.1 Adaptateurs / passerelles

Les passerelles ou les adaptateurs sont la classe de systèmes visant à améliorer l'interopérabilité entre les périphériques IoT. Cette approche utilise un outil intermédiaire appelé médiateur installé dans des passerelles ou des adaptateurs IoT pour converser des protocoles, des données et des normes entre les périphériques d'envoi et de réception. Plusieurs propositions, tant académiques qu'industrielles, portent sur la conception et la normalisation de la passerelle IoT. Ponte [85] présente un cadre permettant l'échange de données entre différents périphériques IoT via différents connecteurs. Zhu et al. [86] proposent une architecture de passerelle IoT utilisant un logiciel programmable dans l'espace utilisateur pour assurer l'interopérabilité entre les protocoles de réseau de capteurs sans fil (WSN) et les réseaux de communication mobile ou Internet. En utilisant la même méthode, les auteurs de [87] présentent une architecture de passerelle s'adaptant aux différences de protocoles de périphériques et de problèmes de sécurité. En tirant parti de la puissance de calcul du smartphone, [88][89] créez une passerelle mobile prenant en charge les mêmes fonctionnalités que la passerelle IoT.

A.2.1.2 API ouvertes

Une API est une interface écrite dans un langage d'abstraction élevé, utilisé pour accéder aux données et fonctions d'une application. Cependant, ces API sont conçues pour être spécifiques à chaque plate-forme ou propriétaires. Pour combler ces manques, HyperCat fournit une spécification permettant une interopérabilité syntaxique entre différentes API et services, décrite sous un format de catalogue [106]. D'autre part, les projets européens Big-IoT ont travaillé sur une API d'interfonctionnement générique qui permet d'accéder aux ressources de toutes les plates-formes IoT existantes. Cette API devrait permettre l'interopérabilité syntaxique et multiplateforme [?]

A.2.1.3 Service orienté architecture

Pour permettre l'interopérabilité entre les périphériques et les plates-formes, les chercheurs ont créé une architecture orientée service au-dessus de la couche réseau [108] afin que les périphériques et les données soient effecivement gérés par des services [108][109]. Les auteurs de [110] ont appliqué les technologies de service Web SOA afin de maximiser le partage et l'interopérabilité des services. En particulier, [111] utilise une approche classique orientée services Web (service Web WS-*) et [112] utilise une approche orientée ressources (services Web REST) afin d'accroître l'interopérabilité syntaxique. Pautasso et al [113] ont comparé les avantages du service Web WS- * et des services Web REST au SOA dans divers cas d'utilisation.

A.2.1.4 Technologies de Semantique web

Actuellement, de nombreuses directions de recherche exploitent les avantages des technologies du Web sémantique dans l'IoT pour parvenir à l'interopérabilité sémantique. Sur le Web sémantique des objets, les ontologies (ou vocabulaires) définissent les concepts ou les relations utilisés pour décrire et représenter un domaine de préoccupation [115]. Plusieurs projets de recherche sur l'IoT ont utilisé des ontologies ou d'autres technologies sémantiques pour améliorer l'interopérabilité dans l'IoT. La sémantique Web des capteurs (SSW) [120] est l'adoption de la technologie des réseaux de capteurs connectés et du Web sémantique. SensorML [121] fourni par l'(OGC) (Open Geospatial Consortium) est une norme basée sur XML qui décrit le capteur du Web. UbiROAD [122] introduit un cadre permettant une interopérabilité sémantique au niveau des données et du protocole fonctionnel. Serrano [123] analyse les défis actuels en matière d'interopérabilité sémantique dans l'IoT.

A.2.1.5 Défis restants

Bien que de nombreuses plateformes IoT, ainsi que des normes, aient émergé pour réaliser l'interopérabilité dans l'IoT, il subsiste des challenges dans ce domaine. Dans cette section, nous présenterons les principaux défis de l'interopérabilité IoT sur la base des solutions examinées.

- *Integration des objets à large echelle*: En raison de l'absence de standard de communication, l'intégration d'un objet hétérogène dans une plateforme IoT, quelle que soit sa technologie de communication, son matériel ou sa configuration, reste un défi majeur. Certaines solutions s'appuient sur une entité réseau, comme une passerelle. Cependant, ces solutions limitent l'évolutivité et la flexibilité nécessaires pour faire face à la croissance de la diversité et de la quantité d' objets IoT.

- *Interoperabilité entre plateforme*: Les plates-formes IoT actuelles fournissent une API ouverte pour accéder à leurs services. Cependant, ces API sont conçues à partir de principes RESTful et de modèle de données specialisé. De plus, l'intégration ne devrait pas nécessiter de changements majeurs dans la plate-forme. De ce fait, l'interopérabilité entre plates-formes reste très difficile.

- *Interoperabilité des données*: Diverses universités, entreprises et entités de normalisation traitent de l'interopérabilité des données IoT en proposant des normes et des description sémantique. Toutefois, cela ne signifie pas que les normes proposées seront acceptées et utilisées largementr. Par conséquent, l'intégration de sources de données hétérogènes afin de maximiser les connaissances exploitables à partir de données IoT constitue un défi de taille.

A.2.2 Fiabilité des données

En règle générale, le nettoyage des données comprend deux étapes principales: (1) Détection des valeurs aberrantes: identification des erreurs ou des événements dans les données, (2) correction des données: correction des erreurs identifiées. Les industriels et les academiques ont pour objectif de nettoyer efficacement les données IoT.

A.2.2.1 Détection des valeurs aberrantes

Methode non supervisée L'une des approches de détection des valeurs aberrantes non supervisées les plus courantes est le concept d'analyse du plus proche voisin. Ces techniques détectent l'anomalie en examinant la distance ou la similarité entre deux instances de données. Les instances de données normales se produisent dans des voisinages denses, tandis que les anomalies se produisent loin de leurs voisins les plus proches [81]. Les auteurs dans [132][133][134] calcule le score d'anomalie à partir de la somme de la distance des k voisins les plus proches [135] Le score d'annomalie est definie comme le nombre de voisins les plus proches dans un rayon de distance d. [136] conçoit une technique d'échantillonnage simple pour réduire la complexité de l'algorithme. Un facteur de résolution hors norme (ROF) a été proposé en [137]. Selon cette méthode, les points sont des valeurs aberrantes ou à l'intérieur d'une grappe, en fonction de la résolution des seuils de distance appliqués.

Methode supervisée Les méthodes de détection d'anomalie supervisée requièrent des données d'apprentissage dans lesquelles sont correctement étiquetées les données correctes et aberrantes. Plusieurs approches utilisant l'apprentissage supervisé ont été proposées telles que MetaCost [143] utilise une approche de reclassement de la classification. L'apprentissage des vecteurs de support est appliqué pour la détection des valeurs aberrantes, comme l'apprentissage en classe [144] et supporte la description des données vectorielles [145]. Une autre approche consiste à utiliser un classificateur de type "transitional machine learning" telque le classificateur de bayes pour classer l'ensemble de données en classes normales et anormales. [146], Classificateur des plus proches voisins [147], arbre de decision [148][149], classificateurs dit "rule based" [150][151] et les classificateur SVM [152][153].

A.2.2.2 Correction des données

Correction de données basée sur le lissage Il s'agit d'une technique simple et légère utilisée pour la correction de données en ligne. Par exemple, une simple moyenne glissante (SMA) [154] calcule la valeur des points actuels à partir de la moyenne des k derniers points. Au lieu d'utiliser une moyenne non pondérée, une alternative est la moyenne mobile pondérée de manière exponentielle (EWMA) [155] dans laquelle les valeurs sont pondérée par un poids en diminution exponentielle dans le temps. Une autre approche appelée lissage SWAB [156] utilise une regression pour corriger en temps réen le flot de données

Techniques de correction de données basées sur des contraintes Les techniques de correction de données basées sur les contraintes corrigent les données en fonction de contraintes données, tout en minimisant les modifications [157][158]. L'algorythme SCREEN [159] fonctionne sous l'hypothèse que la vitesse des changements de données (à savoir la contrainte de vitesse) est contrainte. Sur la base de cette hypothèse, ils proposent une solution permettant au flux de données d'identifier et de corriger les valeurs de «sauts» dans une séquence de données (fenetre de données) avec la contrainte de vitesse tout en minimisant la distance de correction. Cependant, les résultats de la correction reposent fortement sur l'exactitude de l'hypothèse initiale. Conscient de ces limitations, un dernier algorithme appelé réparation minimale itérative (IMR) [160] est proposé. L'idée générale d'un tel algorithme est que la combinaison entre le marquage de certaines observations annormales et la correction itérative basée sur la croissance des fiabilités pourrait améliorer les performances.

A.2.2.3 Défis ouverts

Assurer la qualité des données, en particulier dans le contexte de l'IoT, a été confronté à de nombreux défis. La plupart des solutions sont dédiées à des objectifs spécifiques tels que la détection d'anomalies ou leur netoyage. Il n'existe aucune solution complète permettant d'identifier ou de corriger les erreurs. De plus, les solutions actuelles de nettoyage des données manquent toujours:

- *Scalabilité:* Avec la croissance exponentielle de l'IoT, la quatité données collectées est massive. L'avantage des approches non supervisées est leur légèreté et leur simplicité. Mais, leur précision est très faible. En revanche, les approches supervisées peuvent offrir une grande précision, mais elles nécessitent beaucoup de temps pour former le modèle. Par conséquent, nous avons besoin d'une solution qui garantisse l'évolutivité tout en maintenant une précision élevée.

- *Flexibility:* Les données IoT pourraient être collectées à partir de diverses sources de données (capteurs, devices, étiquettes RFID, par exemple). Les techniques de nettoyage des données doivent être capables d'analyser et de combiner les relations de diverses sources de données afin d'accroître la précision. De plus, les techniques proposées doivent gérer différentes variables décrivant les intérêts de l'utilisateur final. Selon les cas d'usage, l'utilisateur requiert une qualité de détection différente. Cette exigence pose un nouveau défi sur la manière de satisfaire la qualité souhaitée par l'utilisateur.

- *Distinguer erreur et événement:* La principale caractéristique qui manque à toutes les techniques de nettoyage des données est la capacité de distinguer les valeurs aberrantes causées par une erreur de celles causées par des événements. La technique actuelle de nettoyage des données est souvent appliquée pour filtrer les données altérées. Cela signifie que les points détectés sont supprimés comme du bruits inutiles. Malheureusement, les données supprimer peuvent contenir des événements notables, également appelés points de changement. Ces changements se produisent par accident (par exemple, un incendie dans une forêt) ou à la suite d'interventions humaines (par exemple, arrosage de l'arbre). La préservation de ces événements est essentielle pour interpréter le contexte.

A.2.3 Fiabilité des des devices

Dans le contexte IoT, le device IoT doit avoir une durée de vie suffisante pour répondre aux exigences de l'application. Par conséquent, la communauté des chercheurs a accordé une attention particulière à un mécanisme écoénergétique pour les dispositifs IoT. De nombreuses approches sont proposées mais la plupart d'entre elles sont héritées du contexte des réseaux de capteurs sans fil.

A.2.3.1 Travaux connexes

Le nombre d'échantillons pourrait être réduit en exploitant la corrélation dans les données collectées. Les auteurs dans [164] utilisent l'analyse temporelle pour créer un échantillonnage adaptatif pour le scénario de surveillance de la neige. Ils ont proposé un algorithme basé sur un CUSUM modifié [165] qui estime de manière adaptative la fréquence maximale actuelle à partir de la tendance des données historiques. Un algorithme similaire est également proposé dans [83], qui utilise un filtre de Kalman pour calculer les taux d'échantillonnage. La corrélation spatiale est utilisée dans [166] pour proposer un schéma nommé «backcasting». L'idée principale est que la région à forte variation de collecte aura une fréquence de collecte plus élevée que les autres. La même idée est également exploitée dans [167]. Une autre approche appelée "technique d'échantillonnage hiérarchique" est appliquée aux dispositifs IoT, qui embarquent differents types de capteurs. En situation normale, le capteur simple à faible consommation d'énergie est utilisé. Lorsque une situation complexe est détectée, les capteurs complexes utilisant une énergie élevée sont activés pour faire face à cette situation. In [168], cette technique est utilisée pour détecter un scénario d'urgence en cas d'incendie. D'autre part, la prédiction de données est appliquée pour réduire le nombre d'échantillons de données en utilisant des modèles d'apprentissage automatique (par exemple, un modèle linéaire, Naive Bayes). La solution de Ken en [171]utilise une fonction de densité de probabilité comme modèle de base pour prévoir les valeurs futures. Les auteurs de [172] ,sur le même principe utilise un filtre de Kalman comme modèle principal pour les prédictions. Dans [173], un modèle probabiliste dynamique (DPM) est utilisé pour implémenter une vue probabiliste des données échantillonnées disponibles. Pour la série temporelle, PAQ [174] utilise un modèle AR visant à réduire la complexité de calcul. Les atteurs dansn [175] enrichisse le modèle AR pour traiter les données incohérentes et les valeurs aberrantes.

A.2.3.2 Les defis à venir

- *Dimenssion multiple*: L'échantillonnage adaptatif est une technique prometeuse pour atteindre une garnde efficacité énergétique. Cependant, la plupart des solutions proposées ou proposées ne calculent la fréquence optimisée que sur la base d'une caractérisation unique, par exemple dans le temps ou dans l'espace. Nous avons donc besoin d'une solution complète pouvant combiner à la fois le temps et l'espace pour exploiter plusieurs informations en même temps. De plus, les approches nécessitent une commplexité de calcul trop importante pour que leur avantage en terme d'économie d'énergie soit également réduit.
- *Flexibility*: Une autre approche appelée «technique d'échantillonnage hiérarchique» est très adaptée pour traiter des environnement complexes qui nécessitent des capteurs «énergivores». Cette approche est donc très économe en énergie, mais elle est également très dédiée à des applications spécifiques. De plus, le coût pour ajouter un capteur supplémentaire doit être pris en compte.
- *Complexité*: Les approches de prédiction de données nécessitent une énorme quantité de données pour former le modèle, de sorte que le stockage et le traitement de ces données côté devices sont des tâches lourdes pour les devices à ressources limitées. Ils devraient pouvoir distribuer ce calcul sur un réseau de devices plutôt que sur un seul device. De plus, la complexité du modèle de base doit être prise en compte.

A.3 Interoperabilité

A.3.1 Un Framework IoT pour interopérer les devices IoT en se basant sur des connecteurs

Notre solution vise à résoudre les défis liés à *l'intégration de devices à grande échelle*. Ce défi provient du fait que chaque objet IoT, tel qu'un capteur ou un actionneur, fournit différentes interfaces logicielles et une configuration de communication permettant d'échanger des données et des informations de contrôle avec un middleware dans le cloud. De plus, ces interfaces et cette configuration ne sont pas normalisées et changent fréquemment, en particulier dans le cas d'un scénario textit Low Wide Wide Area (LPWAN). Pour récupérer les données de surveillance des devices IoT utilisant une connexion LoRa, nous devons configurer une "callback" HTTP suivant un format spécial défini par un fournisseur de réseau.

Cependant, chaque fournisseur dispose de divers formats de configuration , ce qui entraîne un énorme fossé en termes d'interopérabilité et de stabilité syntaxiques.

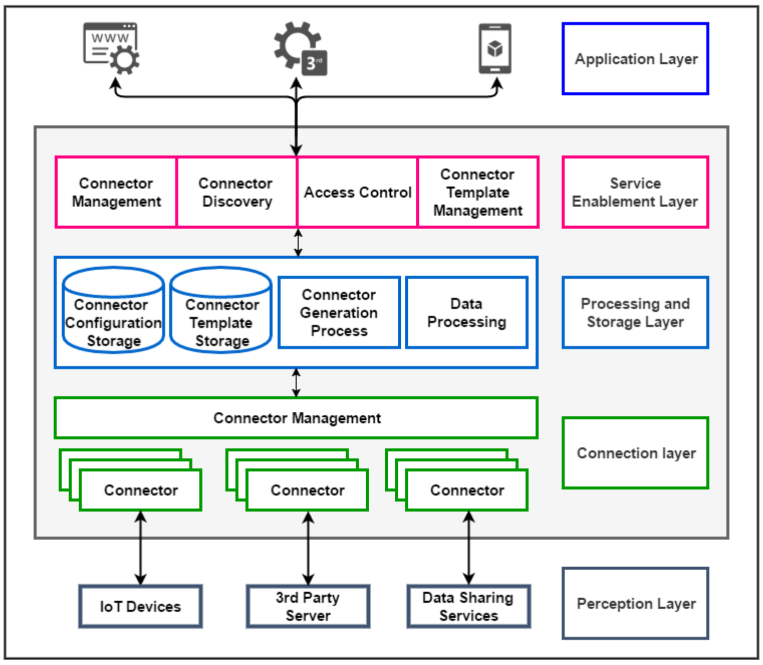


Figure A.1 – L'architecture du framework.

Pour combler cette lacune, de nombreuses approches ont été proposées. Cependant, la plupart limitent le type de capteur pouvant être connecté et demandent une application spécifique installée côté appareil. En outre, l'ajout d'un nouveau capteur aux plates-formes est complexe et nécessite des compétences de programmation avancées. Il n'existe aucun mécanisme permettant de gérer l'évolution rapide des objets IoT en termes d'interface logicielle, de protocole de connectivité et de format de données. En outre, les "frameworks" examinées ne permettent pas à l'utilisateur final d'établir une connexion et de collecter des données à partir de sources de données ouvertes via les protocoles HTTP, MQTT, CoAP ou WS. Pour surmonter ces limitations, nous présentons un nouveau "framework" IoT industriel prenant en charge le processus automatique d'établissement et de configuration de la connectivité hétérogène à l'aide d'un connecteur. En général, le connecteur est un segment de code spécifique qui effectue le processus d'acquisition de données à partir d'un type de connexion spécifique à l'aide de protocoles tels que HTTP, MQTT. Notre infrastructure fournit également des API de gestion permettant d'effectuer des opérations complètes de création, de lecture, de mise à jour et de suppression sur les connecteurs. De plus, le mécanisme d'automatisation de la connectivité aide l'utilisateur final à récupérer rapidement les données de diverses sources de données via les connecteurs. Fig. A.1 décrit le "framework" proposé qui est composé de trois couches différentes décrites ci-dessous.

- **Couche d'activation de service:** Cette couche est composée de plusieurs services Web qui permettent à l'utilisateur final une interaction directe avec le "framework". Les opérations prises en charge sont (1) la découverte du connecteur, (2) les opérations CRUD sur les connecteurs et les modèles de connecteurs, (3) l'activation / désactivation du connecteur et (4) le contrôle d'accès basé sur un jeton de session.

- **Couche de traitement et de stockage:** Cette couche contient les bases de données et les fonctions principales permettant de générer les connecteurs ainsi que les données de pré-traitement. Elle contient également une base de données pour stocker le connecteur généré, le modèle de connecteur et l'état du connecteur.
- **La couche de connexion:** Elle est composée de nombreux connecteurs générés qui ont la charge de se connecter aux objets connectés. Actuellement, notre infrastructure prend en charge deux types de connecteurs, des "Connecteurs entrants" et des "Connecteurs sortants". Ils sont analogues aux concepts de «proxy-in» et de «proxy-out» introduits dans [191]. Chaque type de connecteur prend en charge les connectivités HTTP, MQTT, CoAP et WebSocket (WS) pour récupérer les données des services de capteurs, d'actionneurs, de passerelles et de services Open Data.

Dans notre évaluation en situation réelle, la taille des connecteurs est inférieure à 1 Ko et la mémoire demandée par notre infrastructure n'est que de quelques mégaoctets. De plus, le temps nécessaire pour établir une connexion est très court, aux alentours de 0,003 ms. De fait, notre framework est suffisament léger pour être déployable sur un middle ware dans le cloud, des gateways, mais aussi sur des smartphones ayant des Go de mémoire interne et des processeurs puissants.

A.3.2 Un framework IoT pour maximiser la connaissance des données en utilisant des capteurs virtuels

Au cours des dernières années, nous avons assisté à une explosion de l'Internet of des objets en termes de nombre et de types de périphériques physiques. Cependant, il existe de nombreuses limitations en ce qui concerne leur puissance de calcul des périphériques, leur stockage et leur connexion. Ils affectent de manière significative le traitement des données sur l'appareil. Le traitement centralisé des données IoT s'est révélé difficile pour de nombreux cas d'utilisation nécessitant une réponse en temps réel. Pour surmonter ces limitations, nous proposons un cadre de capteurs virtuels évolutif qui prend en charge la création d'un *logical data-flow* (LDF) en visualisant des capteurs physiques ou des capteurs virtuels personnalisés. Le processus produit des informations de haut niveau à partir des données collectées qui peuvent être facilement interpretées par les machines et les humains. Un *virtual sensor editor* (VSE) sur le Web est également implémenté sur la aprtie haute du framework pour simplifier la création et la configuration du fichier LDF. VSE prend en charge la vérification multiplate-forme et en temps réel du fichier LDF généré. Notre infrastructure prend en charge différents types de capteurs virtuels sous *Infrastructures as a Service* (IaaS) et *Platform as a Service* (PaaS), pour aggréer, cumuler, qualifier le contexte et prédire simplement [202]. Dans le SVFs, les VS sont traités comme des capteurs physiques. Les données de sortie de VS sont stockées dans la base de la même manière que les données physiques. Ainsi, chaque système virtuel contient les données historiques, qui sont précieuses pour une analyse ultérieure des données.
L'objectif principal de notre cadre est de produire des informations de haut niveau à partir des données collectées à l'aide d'un flux de données logique. Cette infrastructure simplifie également la création et la configuration du flux de données logiques en offrant un éditeur de capteur virtuel interactif et de nombreux types d'opérateurs efficaces tels que règle, formule et fonction. La Fig. A.2 décrit l'architecture de la du framework composée de trois couches horizontales: (1) La couche de connexion gère le maintien de la connexion des VSF et de *Sensor Data Service Platform* (SDSP), pour aggréger et traiter les données collectées à partir de capteurs physiques. (2) La couche de traitement contient la base de données et un moteur principal permettant d'exécuter les fonctionnalités de flux de données logiques

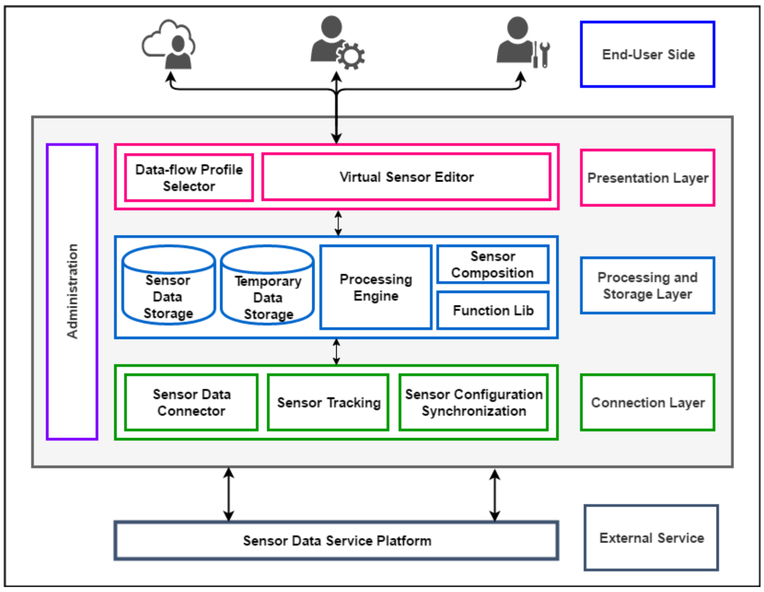


Figure A.2 – Aperçu de l'architecture du framework.

et de capteur virtuel. (3) La couche de présentation supervise le rendu d'une interface Web HTML5 interactive, à savoir l'éditeur de capteurs virtuels. (4) La couche administration joue un rôle dans l'autorisation et la supervision des droits d'accès des utilisateurs sur les capteurs virtuels et les flux de données logiques.

L'efficacité et l'évolutivité de notre proposition ont été démontrées expérimentalement dans un projet industriel de surveillance de réservoirs. Dans notre évaluation, le temps de réponse de notre infrastructure reste stable en dessous d'une seconde, même en cas d'augmentation du nombre de capteurs virtuels simultanés.

A.3.3 Un langage descriptif pour un groupe d'objets

Dans le contexte de Web of Things (WoT), l'utilisateur est capable de créer, de mélanger et de présenter simplement plusieurs objets pour obtenir des informations de haut niveau. Cependant, les recherches actuelles accordent beaucoup plus d'attention à décrire une seule chose. La modélisation et la construction de l'application pour les objets composés constitués de groupes d'objets, à savoir « des Actifs (asset)», sont toujours limitées en raison de l'absence de description et d'un mécanisme d'intégration continue. De plus, le langage de description de périphérique IoT traditionnel directement installé sur le périphérique est fortement limité dans le scénario de déploiement à l'écherlle en raison d'exigences strictes en matière de consommation d'énergie et de coût de fonctionnement. Toutes ces limitations peuvent poser problème dans *IoT-data interoperability*. Dans ce travail, nous introduisons le *WoT based Asset Description* (WoT-AD), un langage descriptif de l'actif visant à atténuer ces limitations. WoT-AD décrit explicitement un groupe d'objets en tant qu'objet homogène pour permettre le mash-up, l'auto-découverte et l'accès simple à leurs ressources, entités et services. Nous fournissons également un cadre léger qui s'intègre parfaitement à WoT-AD pour permettre à WoT de réaliser un scénario de déploiement IoT à l'échelle. Une telle intégration modélise efficacement l'actif, mais simplifie également le développement d'applications de mash-up destinées à aux utilisateurs finaux.

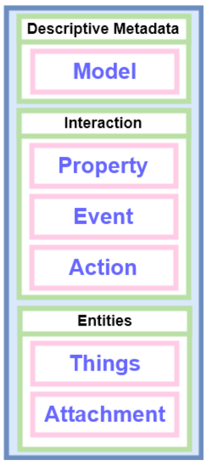


Figure A.3 – Aperçu du modèle d'actif ou Asset.

WoT Asset Description est une description sémantique considérée comme un point d'entrée abstrait d'un groupe d'objets permettant un processus efficace de découverte, d'accès et de gestion. WoT-AD est un concept étendu de la description de W3c Things (W3C-TD), qui utilise un schéma JSON-LD pour décrire les objets simples. La structure WoT-AD comprend trois sections principales, comme illustré dans la figure A.3: (1) La *Descriptive meta-data* décrit les informations générales sur l'actif. (2) Le *interaction model* présente les ressources de l'actif (AR) dans le schéma sémantique. (3) Les *Entities* expriment les objets et les services constituant l'actif ou leurs relations.

Pour maximiser les avantages de WoT-AD, nous proposons un cadre WoT exploitant pleinement les concepts de WoT-AD dans un scénario d'usage IoT à l'échelle. La mise en œuvre initiale de cette architecture est représentée par trois composants principaux répartis sur des couches horizontales, dont les caractéristiques correspondent bien aux principes mentionnés. Le *connector framework* est utilisé au niveau de la couche la plus basse [29] pour gérer les connexions hétérogènes à la fois de périphériques IoT et de sources de données externes. Le second composant est le *virtual sensor framework* [30], un framework IoT permettant de créer le flux de données logique en visualisant les ressources. Le dernier composant est un *Graphic Editor* facilitant le processus de construction et de configuration de l'actif en proposant les actions de glisser-déposer sur l'interface Web HTML5. L'ensemble du cadre est illustré à la figure A.4. Les détails de chaque couche sont décrits ci-dessous:

- **Couche de connexion:** Cette couche gère la connexion des objets LPWAN vers le framework via un connecteur dédié. Ces connecteurs sont simplement créés et gérés par le framework de connecteurs. Au niveau de cette couche, les données collectées des devices sont agrégées et pré-traitées avant d'être transmises à la couche supérieure. Si un nouveau device se connecte au framework, les ressources de ce device seront enregistrées et suivies par des services de suivi de capteurs.

- **Couche de traitement:** Cette couche est considérée comme une partie principale de l'architecture composée de quatre composants principaux: (1) La plate-forme de capteurs virtuels est utilisée pour créer l'Asset en abstrayant et en reliant ensemble les ressources du device. Cette plate-forme est également chargée de mettre à jour les ressources de l'Asset à partir des données collectées (2) Indexées: une fois les ressources de l'Asset créées, elles sont indexées par le système d'indexation. Cela garantit que toutes les ressources sont créées en temps réel et synchronisées avec des

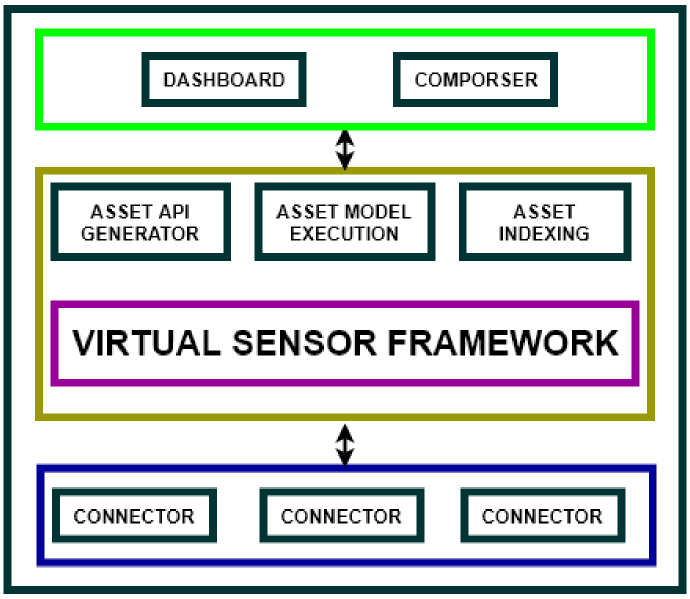


Figure A.4 – Aperçu de l'achitecture de l'actif ou Asset.

device physiques. (3) Le script des Assets est utilisé pour générer la description et API de l'Asset en fonction de ressources définies. (4) L'exécution du modèle de l'Asset convertit le modèle de l'Asset en un flux de données logique [30] pouvant être utilisé par le framework de capteurs virtuels.

- **Couche de Présentation:** Cette couche contient une interface Web interactive HTML 5, à savoir Asset Composer. Cette interface graphique fournit à l'utilisateur plusieurs composants IoT tels que capteur, actionneur, Asset, lien symbolique. L'utilisateur final peut simplement créer son propre Asset par des actions "drag and drop". Un tableau de bord pour gérer l'Asset actuel est également proposé. L'efficacité de la solution conçue a été assurée par le choix et la combinaison de certaines technologies et d'infrastructures IoT éprouvées dans des cas d'utilisation concrets. Pour améliorer et étendre ces travaux, nous optimiserons le scénario appliqué aux bâtiments intelligents.

A.4 Fiabilité dans l'IoT

A.4.1 Une méthode d'apprentissage actif pour la détection d'anomalies

D'erreurs et d'événements est omniprésente dans les séries de données chronologiques, en particulier les lectures de capteurs IoT. La détection d'anomalies au fil des séries temporelles est souvent appliquée pour filtrer les données altérées. Cela signifie que les points détectés sont éliminés comme des bruits inutiles. Malheureusement, les données éliminées peuvent contenir des événements notables, également appelés points de changement. Ces changements se produisent par accident (par exemple, un feu dans une forêt) ou à la suite d'une intervention humaine (par exemple, arrosage de l'arbre). La préservation de ces événements est essentielle pour interpréter le contexte. Comme indiqué dans la section 2.2, les méthodes existantes de détection des anomalies ne permettent pas de distinguer les anomalies représentant des erreurs de données (telles que les lectures incorrectes d'un capteur) des événements notables (telles que l'action d'arrosage dans la surveillance des sols). En outre, la détection complètement automatique des anomalies pourrait ne pas fonctionner correctement sans discrimination entre les anomalies et les points de changement.

Une approche consiste à utiliser une méthode d'apprentissage supervisé [248][249][250]. Cependant, les données étiquetées ne sont pas disponibles en général. Cela nous motive à exploiter une approche interactive. L'objectif est de minimiser l'implication de l'utilisateur tout en garantissant la qualité des résultats du processus. Ceci est réalisé par une méthode d'apprentissage actif qui est utilisé pour obtenir la vérité des utilisateurs. Nos expérimentations montrent que l'étiquetage d'un nombre très limité de points de données peut considérablement améliorer la qualité de la détection. Afin de réduire le nombre d'interactions, nous proposons un algorithme non paramétrique, qui détecte avec précision les anomalies et les points de changement en combinant un nouveau concept de voisinage, à savoir Inverse le plus proche voisin (INN) et une classification probabiliste non supervisée. La philosophie de INN découle de l'observation d'objets dans la réalité. Deux objets sont liés s'ils ont une connexion à double sens. En appliquant ce concept à l'objet de données, l'objet A et l'objet B ont une relation forte si A est le voisin de B et vice-versa, B est le voisin de A. L'exploitation des avantages de l'INN permet de classer le type d'un point de données à l'aide de l'évaluation des scores, calculés à partir des propriétés des INN. L'avantage du concept de l'INN est également démontré dans la détection des anomalies collectives. Si un point de données est identifié comme un point anormal, ses INN sont très anormaux. Notre algorithme propagera le score d'anomalie d'un tel INN pour remonter l'ensemble du pattern d'anomalie. Lorsque l'évaluation d'un point de données appartient à une anomalie collective, son INN sera l'intégralité de cette anomalie, tandis que sa KNN contient des points anormaux et normaux.

Algorithm 2: Anomaly and Change Point Detection

```
Input:  time series X, threshold $\gamma$ (optional)
Output: Error list $Y$, change point list $Z$
1. $\theta \leftarrow$ Candidate(X)
2. Y, Z = [ ]
3. For $x_i$ in $\theta$ do
       $\beta^{(x_i)} \leftarrow$ Score($x_i, X$)
4. $\eth \leftarrow$ Evaluate($\beta$)
5. $CW, Y, Z \leftarrow$ Evaluate Detection($\eth$)
6. Y $\leftarrow$ Score Propagation($\eth$)
7. If min(CW) $\leq \gamma$ then
       Labeling and Go to step 4

   Return $Y, Z$
```

Contrairement aux algorithmes actuelles de détection d'anomalies, qui ciblent la détection ou d'une anomalie unique ou d'une anomalie collective et sont toutes deux vulnérables aux configurations de paramètres. Nous proposons une détection complète, tenant compte à la fois de l'anomalie et du point de changement. En outre, l'utilisation de l' Active Learning permet non seulement d'augmenter considérablement la précision, mais également de réduire la sensibilité aux configurations de paramètres (optimisés). Soit $X = \{x_1, x_2, ..., x_n\}$ qui désigne une série chronologique, où Y et Z sont un ensemble de points d'anomalie et de points de rupture de X respectivement. Algorithm 2 présente les principales étapes de notre proposition, qui prend X en entrée et produit Y et Z, y compris leurs poids de confiance. Notre algorithme permet également aux utilisateurs finaux de configurer le coefficient de confiance souhaité pour assurer la qualité de la détection. Les principales étapes sont décrites ci-dessous:

1. **Estimation du candidat**, dans la ligne 1, génère les candidats potentiels, notés par , à partir des valeurs extrêmes dans les séries temporelles basées sur la dérivée seconde absolue absolue.
2. **Calcul du score**, dans la ligne 3, calcule la métrique du score à partir de l'INN de chaque candidat x_i in θ. Cette métrique comprend le score d'amplitude, le score de corrélation et le score de variance, notés $\beta^{(x_i)}$
3. **Score d'évaluation**, ligne 4, utilise une classification probabiliste pour classer les candidats dans trois classes, notamment les points de changement, les points d'anomalie uniques ou les anomalies analysées collectivement. L'apprentissage actif utilisant le modèle d'incertitude, décrit par l'équation 7.13, est également appliqué. Les points les plus incertains seront appelés et étiquetés pour optimiser le classifieur. Le résultat de cette étape est le coefficient de confiance (CW) (noté $\eth^{(x_i)}$) également appelé score d'anomalie.
4. **Propagation du score d'anomalie**, propage le score d'anomalie vers ses INN au cas où le point d'analyse serait détecté et appartiendrait à une anomalie pattern.
5. **Évaluation de la classification**, est l'étape qui déclenche le processus d'apprentissage actif si le minimum de pondération de confiance est inférieur aux exigences de qualité de l'utilisateur.

Nous évaluons de manière expérimentale la qualité et l'efficacité de notre proposition sur des jeux de données réels et synthétiques. Ces résultats sont également comparés à l'une des approches de détection d'anomalies courantes pour démontrer que: (1) La supériorité de la CABD en qualité de détection pour les deux types (anomalie, point de changement) et en temps d'exécution par rapport à des ensembles de données réels et synthétiques. (2) L'efficacité du concept INN et de l'apprentissage actif dans notre proposition. (3) La CABD pourrait constituer un complément aux algorithmes de réparation des données.

A.4.2 Un algorithme d'échantillonnage économe en énergie

Les techniques d'économie d'énergie d'IoT est essentiel pour la fiabilité des services de l'Internet des objets, en particulier dans le scénario LPWAN, qui requiert de manière rigoureuse une solution économique et à faible consommation d'énergie. La plupart des techniques de conservation de l'énergie supposent généralement que l'acquisition et le traitement des données ont une consommation d'énergie nettement inférieure à celle de la communication. Malheureusement, cette hypothèse n'est pas correcte dans l'IoT, où les capteurs peuvent consommer encore plus d'énergie que la transmission. En outre, la flexibilité et la complexité des techniques de conservation de l'énergie sont encore difficiles (voir section 2.3). Nous proposons un algorithme d'échantillonnage adaptatif permettant d'estimer en temps réél les fréquences d'échantillonnage optimales pour les capteurs en fonction des modifications des données collectées. Compte tenu des besoins d' économies d'énergie des utilisateurs, notre algorithme pourrait réduire la consommation d'énergie des capteurs tout en maintenant une très grande précision des données collectées.

Contrairement aux algorithmes d'échantillonnage adaptatif existants qui sont vulnérable aux paramètres de configuration et nécessitent une acquisition fréquente des données, nous proposons une solution légère d'algorithm d'échantillonnage conçue pour optimiser l'utilisation de la puissance des devices restreint dans l'Internet des objets. Compte tenu du niveau d'économie d'énergie souhaité, l'algorithme optimise la précision des données de l'échantillon pour ce niveau. L'idée générale de notre algorithme est d'adapter dynamiquement la fréquence d'échantillonnage aux changements d'observation. De toute évidence, une fréquence plus élevée est fortement préférée à une prise de conscience du contexte lorsqu'il y a des changements significatifs dans l'observation. Par exemple, dans les services d'alerte incendie en forêt, l'augmentation soudaine de la température est considérée comme un événement notable. L'augmentation de la fréquence peut aider à approfondir les investigations sur de tels événements. En revanche, si la valeur observée fluctue à peine, une diminution de fréquence est souhaitée pour économiser de l'énergie lors de l'échantillonnage, du traitement et de la transmission des données.

Pour archiver ces hypothèses, une fonction sigmoïde améliorée est utilisée pour adapter rapidement la fréquence d'échantillonnage, décrite comme suit:

$$f_{change} = n + \frac{1-n}{1+e^{-n*D}} \tag{A.1}$$

with:

$$D = \frac{\triangle\theta_i(X_i) - \frac{n+1}{2} * (\frac{1}{N}\sum_{j=i-N}^{i} \triangle\theta_i(X_j))}{\frac{1}{N}\sum_{j=i-N}^{i} \triangle\theta_i(X_j)} \tag{A.2}$$

Dans l'équation A.1, n est le besoin d'économie d'énergie de l'utilisateur; et D présente le changement soudain des données à venir en comparant avec les N données les plus récentes basées sur une fenêtre glissante. Il est raisonnable de comparer le changement de la différence absolue entre la première différence absolue de Xi et la valeur moyenne de cette différence des derniers N points de données. Si D est suffisamment grand, cela signifie que le changement actuel déborde par rapport à l'histoire récente. Ensuite, la fréquence d'échantillonnage est adaptée à ce changement. En revanche, si la valeur de D est négative, ce qui signifie que les valeurs ne sont pas suffisamment grandes, la fréquence d'échantillonnage peut être réduite pour économiser les ressources du dispositif.

En tant que résultat de la fonction sigmoïde étendue, la valeur théorique de la nouvelle fréquence est limitée dans une plage allant de la fréquence standard (supposée égale à 1) au désir de l'utilisateur (n), c'est-à-dire quelle que soit la valeur de D, nouvelle fréquence indiquée par y (D) est toujours dans la limite définie. Cette barrière est vitale dans un réseau contraint tel que LPWAN, SIGFOX alors que l'intervalle minimum d'envoi des messages est strictement limitée. La présentation de la fonction y (D) est illustrée à la figure 1. On constate que la valeur de y varie fortement lorsque la valeur D est comprise entre -1 et 1, c'est-à-dire que le changement des dernières données échantillonnées est supérieur de $\frac{n-1}{2}$ à $\frac{n+3}{2}$ fois à la moyenne de ce changement d'historique. Pour la valeur restante de D, le y (D) converge vers le bord de sa limite. Par conséquent, la nouvelle fréquence d'échantillonnage pour la prochaine itération est capable de s'adapter au changement des données de dernière minute et aux limites souhaitées, ce qui satisfait les besoins d'un algorithme de fréquence efficace ainsi que de conservation de l'énergie.

Des expériences pratiques sur le scénario réel ont montré que l'algorithme proposé peut réduire jusqu'à 4 fois le nombre d'échantillons acquis par rapport à une approche classique à taux fixe avec une erreur extrêmement faible, autour de 4,37%.

A.5 Conclusion

Nous avons assisté à l'explosion de l'internet des objets et à ses impacts positifs sur plusieurs aspects de la vie réelle. Toutefois, dans le déploiement à grande échelle, plusieurs problèmes liés à l'IoT ont été résolus. La plupart d'entre eux concernent la sécurité, la confidentialité, l'interopérabilité et la fiabilité. Les problèmes sont dus au fait que l'IoT est généralement constitué de petits objets de capacités largement distribuées et limitées en termes de stockage et de puissance de calcul. De plus, les données brutes collectées à partir des objets IoT sont massives, hétérogènes et contiennent une grande quantité d'informations redondantes.

Pour s'attaquer à ces problèmes, cette thèse a proposé des solutions pour améliorer l'interopérabilité et la fiabilité. Cette thèse permet d'atteindre les objectifs suivants:

- *Identifier les états actuels et les défis de l'Internet des objets basé sur le cloud:* Dans le cadre de cette thèse, nous soulignons simplement des questions telles que l'interopérabilité, la fiabilité et la facilité de traitement des données.
- *Proposer un framework IoT pour atteindre l'interopérabilité au niveau de la communication entre devices:* La solution proposée est un framework IoT innovant qui pourrait automatiser le processus de création de connecteur middleware basé des Objets dans le cloud utilisés dans des environnements industriels. De cette manière, notre solution peut résoudre efficacement au problème lié à l'intégration de device à grande échelle.
- *Introduire un framework IoT pour maximiser les connaissances utilisables à l'aide de capteurs virtuels:* En exploitant les avantages du capteur virtuel, notre travail vise à résoudre le problème d'interopérabilité des données IoT. Le cadre proposé pourrait intégrer diverses sources de données IoT pour produire des informations de haut niveau à partir des données collectées.
- *Présenter un langage descriptif sémantique pour un groupe d'objets:* Dans le but de résoudre le problème d'interopérabilité entre plateformes croisées, nous introduisons un nouvel objet composé sémantiquement présenté par langage descriptif dans le scénario de Massive IoT. Nous proposons également un cadre léger Web of Things, entièrement intégré pour maximiser l'interopérabilité.
- *Proposer une méthode d'apprentissage actif pour la détection des erreurs et des événements dans les séries temporelles:* Notre méthode détecte efficacement les erreurs et les événements avec un algorithme unique basé sur l'apprentissage actif. En outre, la qualité de la détection est contrôlée par la confiance de la classification, qui est ensuite utilisée comme condition de terminaison de la procédure d'apprentissage actif.
- *Proposer un algorithme d'échantillonnage économe en énergie*: Résoudre les problèmes liés à la flexibilité et à la complexité dans un recherche d'éfficacité energetique des dispositif IoT.. L'algorithme proposé minimise la consommation d'énergie en estimant en temps réel la fréquence optimale de collecte des données sur la base de données historiques.

Bénéfice des solutions - Application au système et projets réels: Toutes les solutions proposées ont été mises en œuvre et fonctionnent dans une plate-forme IoT basée sur le cloud d'un start-up . Cette plate-forme a pour objectif de fournir des services cloud basés sur les technologies Internet of Things afin de résoudre des problèmes de la vie courante. Afin de gérer la connexion entre divers objets IoT, notre solution relative à l'interopérabilité au niveau de la communication entre devices est déployée au niveau de la couche de communication de ce framework. Ensuite, la deuxième solution utilisant des capteurs virtuels pour optimiser les connaissances acquises grâce à ces appareils est également déployée. En outre, un algorithme de détection des erreurs et des événements est mis en œuvre pour maintenir une qualité élevée des données. D'autre part, l'algorithme d'échantillonnage à efficacité énergétique est implémenté sur tous les dispositifs IoT, en particulier les appareils à ressources limitées.

Appendix B
List of Publications

The results obtained in this dissertation have been published/submitted in:

Publications:

1. L. Kim-Hung, S. K. Datta, C. Bonnet, F. Hamon and A. Boudonne, "An industrial IoT framework to simplify connection process using system-generated connector," 2017 IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI), Modena, 2017, pp. 1-6.
2. L. Kim-Hung, S. K. Datta, C. Bonnet, F. Hamon and A. Boudonne, "A scalable IoT framework to design logical data flow using virtual sensor," 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Rome, 2017, pp. 1-7.
3. L. Kim-Hung, S. K. Datta, C. Bonnet and F. Hamon, "WoT-AD: A Descriptive Language for Group of Things in Massive IoT," 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), Submitted.

Patents:

1. Procede pour detecter distinctement des anomalies isolees, des anomalies collectives et des points de ruptures et dans une serie temporelle de mesures capteur, 2018, Patent No L. 612-2 R.612-8, France
2. Procede pour reduire la consommation d‘energie ainsi que la frequence de mesure et de transmission d’un capteur connecte, 2018, Submitted.

Appendix C
Indexes

Index

Bibliography

[1] A. Bassi and G. Horn, "Internet of things in 2020: A roadmap for the future," *European Commission: Information Society and Media*, vol. 22, pp. 97–114, 2008.

[2] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] B. David, C. Yin, Y. Zhou, T. Xu, B. Zhang, H. Jin, and R. Chalon, "Smart-city: Problematics, techniques and case studies," in *Computing Technology and Information Management (ICCM), 2012 8th International Conference on*, vol. 1. IEEE, 2012, pp. 168–174.

[4] D. Giusto, A. Iera, G. Morabito, and L. Atzori, Eds., *The Internet of Things.* Springer New York, 2010. [Online]. Available: https://doi.org/10.1007/978-1-4419-1674-7

[5] R. Yuan, L. Shumin, and Y. Baogang, "Value chain oriented rfid system framework and enterprise application," *Beijing: Science*, 2007.

[6] J. Al-Kassab, P. Blome, G. Wolfram, F. Thiesse, and E. Fleisch, "Rfid in the apparel retail industry: A case study from galeria kaufhof galeria kaufhof," in *Unique radio innovation for the 21st century.* Springer, 2011, pp. 281–308.

[7] S. Intelligence, "Six technologies with potential impacts on us interests out to 2025," *National Intelligent Concil, Tech. Rep*, 2008.

[8] D. Floyer, "Defining and sizing the industrial internet," *Wikibon, Marlborough, MA, USA*, 2013.

[9] "The internet of things [infographic]," https://blogs.cisco.com/diversity/the-internet-of-things-infographic, (Accessed on 12/13/2018).

[10] I. Lee and K. Lee, "The internet of things (iot): Applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015.

[11] L. M. Kaufman, "Data security in the world of cloud computing," *IEEE Security & Privacy*, vol. 7, no. 4, 2009.

[12] G. C. Fox, S. Kamburugamuve, and R. D. Hartman, "Architecture and measured characteristics of a cloud based internet of things," in *2012 international conference on Collaboration Technologies and Systems (CTS).* IEEE, 2012, pp. 6–12.

[13] S. K. Dash, S. Mohapatra, and P. K. Pattnaik, "A survey on applications of wireless sensor network using cloud computing," *International Journal of Computer Science & Emerging Technologies*, vol. 1, no. 4, pp. 50–55, 2010.

[14] G. Suciu, A. Vulpe, S. Halunga, O. Fratu, G. Todoran, and V. Suciu, "Smart cities built on resilient cloud computing and secure internet of things," in *Control Systems and Computer Science (CSCS), 2013 19th International Conference on.* IEEE, 2013, pp. 513–518.

[15] S. Aguzzi, D. Bradshaw, M. Canning, M. Cansfield, P. Carter, G. Cattaneo, S. Gusmeroli, G. Micheletti, D. Rotondi, and R. Stevens, "Definition of a research and innovation policy leveraging cloud computing and iot combination," *Final Report, European Commission, SMART*, vol. 37, p. 2013, 2013.

[16] D. Bandyopadhyay and J. Sen, "Internet of things: Applications and challenges in technology and standardization," *Wireless Personal Communications*, vol. 58, no. 1, pp. 49–69, 2011.

[17] H. Veer and A. Wiles, "Achieving technical interoperability-the etsi approach, european telecommunications standards institute," 2008.

[18] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless communications and mobile computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[19] B. Ahlgren, M. Hidell, and E. C.-H. Ngai, "Internet of things for smart cities: Interoperability and open data," *IEEE Internet Computing*, no. 6, pp. 52–56, 2016.

[20] A. Botta, W. de Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: A survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, mar 2016. [Online]. Available: https://doi.org/10.1016/j.future.2015.09.021

[21] A. Gluhak, S. Krco, M. Nati, D. Pfisterer, N. Mitton, and T. Razafindralambo, "A survey on facilities for experimental internet of things research," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 58–67, 2011.

[22] J. W. Branch, C. Giannella, B. Szymanski, R. Wolff, and H. Kargupta, "In-network outlier detection in wireless sensor networks," *Knowledge and information systems*, vol. 34, no. 1, pp. 23–54, 2013.

[23] D. Zeng, S. Guo, and Z. Cheng, "The web of things: A survey," *Journal of Communications*, vol. 6, no. 6, pp. 424–438, 2011.

[24] K. Lee, D. Murray, D. Hughes, and W. Joosen, "Extending sensor networks into the cloud using amazon web services," in *Networked Embedded Systems for Enterprise Applications (NESEA), 2010 IEEE International Conference on.* IEEE, 2010, pp. 1–7.

[25] K. Mikhaylov, J. Petaejaejaervi, and T. Haenninen, "Analysis of capacity and scalability of the lora low power wide area network technology," in *European Wireless 2016; 22th European Wireless Conference; Proceedings of.* VDE, 2016, pp. 1–6.

[26] C. Alippi, G. Anastasi, M. Di Francesco, and M. Roveri, "An adaptive sampling algorithm for effective energy management in wireless sensor networks with energy-hungry sensors," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 2, pp. 335–344, 2010.

[27] A. Baranov, D. Spirjakin, S. Akbari, and A. Somov, "Optimization of power consumption for gas sensor nodes: A survey," *Sensors and Actuators A: Physical*, vol. 233, pp. 279–289, 2015.

[28] W. W. W. Consortium *et al.*, "Json-ld 1.0: a json-based serialization for linked data," 2014.

[29] L. Kim-Hung, S. K. Datta, C. Bonnet, F. Hamon, and A. Boudonne, "An industrial iot framework to simplify connection process using system-generated connector," in *Research and Technologies for Society and Industry (RTSI), 2017 IEEE 3rd International Forum on.* IEEE, 2017, pp. 1–6.

[30] ——, "A scalable iot framework to design logical data flow using virtual sensor," in *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob).* IEEE, 2017, pp. 1–7.

[31] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of things (iot): A literature review," *Journal of Computer and Communications*, vol. 3, no. 05, p. 164, 2015.

[32] P. P. Ray, "A survey on internet of things architectures," *Journal of King Saud University-Computer and Information Sciences*, vol. 30, no. 3, pp. 291–319, 2018.

[33] R. Minerva, A. Biru, and D. Rotondi, "Towards a definition of the internet of things (iot)," *IEEE Internet Initiative*, vol. 1, pp. 1–86, 2015.

[34] D. Guinard and V. Trifa, *Building the web of things: with examples in node. js and raspberry pi.* Manning Publications Co., 2016.

[35] D. Pfisterer, K. Romer, D. Bimschas, O. Kleine, R. Mietz, C. Truong, H. Hasemann, A. Kröller, M. Pagel, M. Hauswirth *et al.*, "Spitfire: toward a semantic web of things," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 40–48, 2011.

[36] A. J. Jara, A. C. Olivieri, Y. Bocchi, M. Jung, W. Kastner, and A. F. Skarmeta, "Semantic web of things: an analysis of the application semantics for the iot moving towards the iot convergence," *International Journal of Web and Grid Services*, vol. 10, no. 2-3, pp. 244–272, 2014.

[37] Northstream, "Massive IoT : different technologies for different needs," Tech. Rep., 2017.

[38] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low power wide area networks: An overview," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 855–873, 2017.

[39] X. Liu and O. Baiocchi, "A comparison of the definitions for smart sensors, smart objects and Things in IoT," *7th IEEE Annual Information Technology, Electronics and Mobile Communication Conference, IEEE IEMCON 2016*, pp. 1–4, 2016.

[40] I. G. Smith, *The Internet of things 2012: new horizons.* CASAGRAS2, 2012.

[41] J. Voas, "Networks of 'things'(nist special publication 800-183)," *National Institute of Standards and Technology*, vol. 30, p. 30, 2016.

[42] Z. Shelby and C. Bormann, *6LoWPAN: The wireless embedded Internet.* John Wiley & Sons, 2011, vol. 43.

[43] G. Mulligan, "The 6lowpan architecture," in *Proceedings of the 4th Workshop on Embedded Networked Sensors*, ser. EmNets '07. New York, NY, USA: ACM, 2007, pp. 78–82. [Online]. Available: http://doi.acm.org/10.1145/1278972.1278992

[44] "adamdunkels/uip: The historical uip sources," https://github.com/adamdunkels/uip, (Accessed on 12/18/2018).

[45] Z. Shelby, J. Riihijärvi, O. Raivio, and O. Mähönen, "Nanoip," 2003.

[46] K. Pister and L. Doherty, "Tsmp: Time synchronized mesh protocol," *IASTED Distributed Sensor Networks*, vol. 391, p. 398, 2008.

[47] S. Cheshire and M. Krochmal, "Multicast dns," Tech. Rep., 2013.

[48] "Hypercat developer," https://hypercatiot.github.io/, (Accessed on 12/18/2018).

[49] "Rfc 6970 - universal plug and play (upnp) internet gateway device - port control protocol interworking function (igd-pcp iwf)," https://tools.ietf.org/html/rfc6970, (Accessed on 12/18/2018).

[50] M. Version, "3.1. 1," *Edited by Andrew Banks and Rahul Gupta*, vol. 10, 2014.

[51] "Rfc 7252 - the constrained application protocol (coap)," https://tools.ietf.org/html/rfc7252, (Accessed on 12/19/2018).

[52] W. Colitti, K. Steenhaut, and N. De Caro, "Integrating wireless sensor networks with the web," *Extending the Internet to Low power and Lossy Networks (IP+ SN 2011)*, 2011.

[53] J. O'Hara, "Toward a commodity enterprise middleware," *Queue*, vol. 5, no. 4, pp. 48–55, 2007.

[54] "starksm64/iotresearch: Research on what makes up iot," https://github.com/starksm64/iotresearch, (Accessed on 12/19/2018).

[55] S. Grant, "3gpp low power wide area technologies-gsma white paper," *gsma. com*, 2016.

[56] "Lora gateways and concentrators | loriot.io," https://www.loriot.io/lorawan.html, (Accessed on 12/19/2018).

[57] "Lorawan r1.0 open standard released for the iot | business wire," https://www.businesswire.com/news/home/20150616006550/en/LoRaWAN-R1.0-Open-Standard-Released-IoT, (Accessed on 12/19/2018).

[58] S. Bandyopadhyay, M. Sengupta, S. Maiti, and S. Dutta, "A survey of middleware for internet of things," in *Recent trends in wireless and mobile networks.* Springer, 2011, pp. 288–296.

[59] "Tinydb: A declarative database for sensor networks," http://telegraph.cs.berkeley.edu/tinydb/, (Accessed on 12/20/2018).

[60] B. Guo, D. Zhang, and Z. Wang, "Living with internet of things: The emergence of embedded intelligence," in *2011 IEEE International Conferences on Internet of Things, and Cyber, Physical and Social Computing.* IEEE, 2011, pp. 297–304.

[61] K. Modukari, S. Hariri, N. V. Chalfoun, and M. Yousif, "Autonomous middleware framework for sensor networks," *International Journal of Pervasive Computing and Communications*, vol. 1, no. 4, pp. 337–345, 2005.

[62] A. Barros and D. Oberle, "Handbook of service description," *USDL and Its Methods*, 2012.

[63] P. Mell, T. Grance *et al.*, "The nist definition of cloud computing," 2011.

[64] J. Spillner, J. MüLler, and A. Schill, "Creating optimal cloud storage systems," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 1062–1072, 2013.

[65] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "On the integration of cloud computing and internet of things," in *Future internet of things and cloud (FiCloud), 2014 international conference on.* IEEE, 2014, pp. 23–30.

[66] M. M. Gomes, R. d. R. Righi, and C. A. da Costa, "Future directions for providing better iot infrastructure," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication.* ACM, 2014, pp. 51–54.

[67] N. Alhakbani, M. M. Hassan, M. A. Hossain, and M. Alnuem, "A framework of adaptive interaction support in cloud-based internet of things (iot) environment," in *International Conference on Internet and Distributed Computing Systems.* Springer, 2014, pp. 136–146.

[68] P. Parwekar, "From internet of things towards cloud of things," in *Computer and Communication Technology (ICCCT), 2011 2nd International Conference on.* IEEE, 2011, pp. 329–333.

[69] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016.

[70] J. Radatz, A. Geraci, and F. Katki, "Ieee standard glossary of software engineering terminology," *IEEE Std*, vol. 610121990, no. 121990, p. 3, 1990.

[71] K. L. Morse, M. D. Petty, P. Reynolds, W. Waite, and P. Zimmerman, "Findings and recommendations from the 2003 composable mission space environments workshop," in *proceedings of the 2004 spring simulation interoperability workshop, Arlington VA*, 2004, pp. 313–323.

[72] I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, and M. Guizani, "Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges," *IEEE wireless communications*, vol. 24, no. 3, pp. 10–16, 2017.

[73] O. Hahm, E. Baccelli, H. Petersen, and N. Tsiftes, "Operating systems for low-end devices in the internet of things: a survey," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 720–734, 2016.

[74] O. Bello, S. Zeadally, and M. Badra, "Network layer inter-operation of device-to-device communication technologies in internet of things (iot)," *Ad Hoc Networks*, vol. 57, pp. 52–62, 2017.

[75] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[76] E. Borgia, "The internet of things vision: Key features, applications and open issues," *Computer Communications*, vol. 54, pp. 1–31, 2014.

[77] M. Bauer, J. Davies, M. Girod-Genet, and M. Underwood, "Semantic interoperability for the web of things," *Available via Research Gate. Accessed November*, vol. 27, p. 2017, 2016.

[78] N. Javed and T. Wolf, "Automated sensor verification using outlier detection in the internet of things," in *Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on.* IEEE, 2012, pp. 291–296.

[79] M. E. Otey, S. Parthasarathy, and A. Ghoting, "Fast lightweight outlier detection in mixed-attribute data," *Techincal Report, OSU–CISRC–6/05–TR43*, 2005.

[80] Y. Zhang, N. Meratnia, and P. J. Havinga, "Outlier detection techniques for wireless sensor networks: A survey." *IEEE Communications Surveys and Tutorials*, vol. 12, no. 2, pp. 159–170, 2010.

[81] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[82] I. Dietrich and F. Dressler, "On the lifetime of wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 5, no. 1, p. 5, 2009.

[83] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad hoc networks*, vol. 7, no. 3, pp. 537–568, 2009.

[84] A. El-Sayed, V. Roca, and L. Mathy, "A survey of proposals for an alternative group communication service," *IEEE network*, vol. 17, no. 1, pp. 46–51, 2003.

[85] "Ponte - m2m bridge framework for rest developers | the eclipse foundation," http://www.eclipse.org/proposals/technology.ponte/, (Accessed on 12/29/2018).

[86] Q. Zhu, R. Wang, Q. Chen, Y. Liu, and W. Qin, "Iot gateway: Bridgingwireless sensor networks into internet of things," in *Embedded and Ubiquitous Computing (EUC), 2010 IEEE/IFIP 8th International Conference on.* Ieee, 2010, pp. 347–352.

[87] R. Fantacci, T. Pecorella, R. Viti, and C. Carlini, "Short paper: Overcoming iot fragmentation through standard gateway architecture," in *2014 IEEE World Forum on Internet of Things (WF-IoT).* IEEE, 2014, pp. 181–182.

[88] C. Pereira, A. Pinto, A. Aguiar, P. Rocha, F. Santiago, and J. Sousa, "Iot interoperability for actuating applications through standardised m2m communications," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2016 IEEE 17th International Symposium on A.* IEEE, 2016, pp. 1–6.

[89] G. Aloi, G. Caliciuri, G. Fortino, R. Gravina, P. Pace, W. Russo, and C. Savaglio, "Enabling iot interoperability through opportunistic smartphone-based mobile gateways," *Journal of Network and Computer Applications*, vol. 81, pp. 74–84, 2017.

[90] Á. Asensio, Á. Marco, R. Blasco, and R. Casas, "Protocol and architecture to bring things into internet of things," *International Journal of Distributed Sensor Networks*, vol. 10, no. 4, p. 158252, 2014.

[91] J. Hoebeke, E. De Poorter, S. Bouckaert, I. Moerman, and P. Demeester, "Managed ecosystems of networked objects," *Wireless Personal Communications*, vol. 58, no. 1, pp. 125–143, 2011.

[92] I. Ishaq, J. Hoebeke, I. Moerman, and P. Demeester, "Internet of things virtual networks: Bringing network virtualization to resource-constrained devices," in *Green Computing and Communications (GreenCom), 2012 IEEE International Conference On.* IEEE, 2012, pp. 293–300.

[93] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[94] N. Bizanis and F. A. Kuipers, "Sdn and virtualization solutions for the internet of things: A survey," *IEEE Access*, vol. 4, pp. 5591–5606, 2016.

[95] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the internet-of-things," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, 2014, pp. 1–9.

[96] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, "Software-defined networking (sdn): Layers and architecture terminology," Tech. Rep., 2015.

[97] S. Bera, S. Misra, and A. V. Vasilakos, "Software-defined networking for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1994–2008, Dec 2017.

[98] J. Li, E. Altman, and C. Touati, "A general sdn-based iot framework with nvf implementation," *ZTE communications*, vol. 13, no. 3, pp. 42–45, 2015.

[99] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: a survey," *IEEE Communications Magazine*, vol. 51, no. 11, pp. 24–31, 2013.

[100] C. Prazeres and M. Serrano, "Soft-iot: Self-organizing fog of things," in *Advanced Information Networking and Applications Workshops (WAINA), 2016 30th International Conference on*. IEEE, 2016, pp. 803–808.

[101] A. Gyrard, M. Serrano, and P. Patel, "Building interoperable and cross-domain semantic web of things applications," in *Managing the Web of Things*. Elsevier, 2017, pp. 305–324.

[102] M. Aazam and E. Huh, "Fog computing and smart gateway based communication for cloud of things," in *2014 International Conference on Future Internet of Things and Cloud*, Aug 2014, pp. 464–470.

[103] S. Cirani, G. Ferrari, N. Iotti, and M. Picone, "The iot hub: a fog node for seamless management of heterogeneous connected smart objects," in *2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking - Workshops (SECON Workshops)*, June 2015, pp. 1–6.

[104] S. K. Datta and C. Bonnet, "An edge computing architecture integrating virtual iot devices," in *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*, Oct 2017, pp. 1–3.

[105] M. Díaz, C. Martín, and B. Rubio, "State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing," *Journal of Network and Computer applications*, vol. 67, pp. 99–117, 2016.

[106] R. Lea, "Hypercat: an iot interoperability specification," 2013.

[107] "Big iot – bridging the interoperability gap of the internet of things," http://big-iot.eu/, (Accessed on 12/28/2018).

[108] S. Vinoski, "Integration with web services," *IEEE internet computing*, vol. 7, no. 6, pp. 75–77, 2003.

[109] S. Li, G. Oikonomou, T. Tryfonas, T. M. Chen, and L. Da Xu, “A distributed consensus algorithm for decision making in service-oriented internet of things,” *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1461–1468, 2014.

[110] M. P. Papazoglou and W.-J. Van Den Heuvel, “Service oriented architectures: approaches, technologies and research issues,” *The VLDB journal*, vol. 16, no. 3, pp. 389–415, 2007.

[111] S. Alam and J. Noll, “A semantic enhanced service proxy framework for internet of things,” in *Proceedings of the 2010 IEEE/ACM Int’l Conference on Green Computing and Communications & Int’l Conference on Cyber, Physical and Social Computing.* IEEE Computer Society, 2010, pp. 488–495.

[112] P. Varga, F. Blomstedt, L. L. Ferreira, J. Eliasson, M. Johansson, J. Delsing, and I. M. de Soria, “Making system of systems interoperable–the core components of the arrowhead framework,” *Journal of Network and Computer Applications*, vol. 81, pp. 85–95, 2017.

[113] C. Pautasso, O. Zimmermann, and F. Leymann, “Restful web services vs. big’web services: making the right architectural decision,” in *Proceedings of the 17th international conference on World Wide Web.* ACM, 2008, pp. 805–814.

[114] F. Scioscia and M. Ruta, “Building a semantic web of things: issues and perspectives in information compression,” in *Semantic Computing, 2009. ICSC’09. IEEE International Conference on.* IEEE, 2009, pp. 589–594.

[115] D. Fensel, “Ontologies,” in *Ontologies.* Springer, 2001, pp. 11–18.

[116] M. Compton, P. Barnaghi, L. Bermudez, R. GarcíA-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. Herzog *et al.*, “The ssn ontology of the w3c semantic sensor network incubator group,” *Web semantics: science, services and agents on the World Wide Web*, vol. 17, pp. 25–32, 2012.

[117] L. Daniele, F. den Hartog, and J. Roes, “Created in close interaction with the industry: the smart appliances reference (saref) ontology,” in *International Workshop Formal Ontologies Meet Industries.* Springer, 2015, pp. 100–112.

[118] J. Soldatos, N. Kefalakis, M. Hauswirth, M. Serrano, J.-P. Calbimonte, M. Riahi, K. Aberer, P. P. Jayaraman, A. Zaslavsky, I. P. Žarko *et al.*, “Openiot: Open source internet-of-things in the cloud,” in *Interoperability and open-source solutions for the internet of things.* Springer, 2015, pp. 13–25.

[119] M. Ganzha, M. Paprzycki, W. Pawłowski, P. Szmeja, and K. Wasielewska, “Semantic interoperability in the internet of things: An overview from the inter-iot perspective,” *Journal of Network and Computer Applications*, vol. 81, pp. 111–124, 2017.

[120] A. Sheth, C. Henson, and S. S. Sahoo, “Semantic sensor web,” *IEEE Internet computing*, vol. 12, no. 4, 2008.

[121] M. Botts and A. Robin, “Opengis sensor model language (sensorml) implementation specification,” *OpenGIS Implementation Specification OGC*, vol. 7, no. 000, 2007.

[122] V. Terziyan, O. Kaykova, and D. Zhovtobryukh, “Ubiroad: Semantic middleware for context-aware smart road environments,” in *Internet and web applications and services (iciw), 2010 fifth international conference on.* IEEE, 2010, pp. 295–302.

[123] A. Gyrard and M. Serrano, "Connected smart cities: Interoperability with seg 3.0 for the internet of things," in *Advanced Information Networking and Applications Workshops (WAINA), 2016 30th International Conference on.* IEEE, 2016, pp. 796–802.

[124] S. De, P. Barnaghi, M. Bauer, and S. Meissner, "Service modelling for the internet of things," in *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on.* IEEE, 2011, pp. 949–955.

[125] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," *SIGMOD Rec.*, vol. 29, no. 2, pp. 93–104, May 2000. [Online]. Available: http://doi.acm.org/10.1145/335191.335388

[126] J. Tang, Z. Chen, A. W.-c. Fu, and D. Cheung, "A robust outlier detection scheme for large data sets," in *In 6th Pacific-Asia Conf. on Knowledge Discovery and Data Mining.* Citeseer, 2001.

[127] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Loop: local outlier probabilities," in *Proceedings of the 18th ACM conference on Information and knowledge management.* ACM, 2009, pp. 1649–1652.

[128] S. Boriah, V. Chandola, and V. Kumar, "Similarity measures for categorical data: A comparative evaluation," in *Proceedings of the 2008 SIAM International Conference on Data Mining.* SIAM, 2008, pp. 243–254.

[129] V. Chandola, S. Boriah, and V. Kumar, "Understanding categorical similarity measures for outlier detection," *Technical report 08–008, University of Minnesota*, 2008.

[130] S. Byers and A. E. Raftery, "Nearest-neighbor clutter removal for estimating features in spatial point processes," *Journal of the American Statistical Association*, vol. 93, no. 442, pp. 577–584, 1998.

[131] S. E. Guttormsson, R. Marks, M. El-Sharkawi, and I. Kerszenbaum, "Elliptical novelty grouping for on-line short-turn detection of excited running rotors," *IEEE Transactions on Energy Conversion*, vol. 14, no. 1, pp. 16–22, 1999.

[132] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, "A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data," *Applications of data mining in computer security*, vol. 6, pp. 77–102, 2002.

[133] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," in *European Conference on Principles of Data Mining and Knowledge Discovery.* Springer, 2002, pp. 15–27.

[134] J. Zhang and H. Wang, "Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance," *Knowledge and information systems*, vol. 10, no. 3, pp. 333–355, 2006.

[135] E. M. Knorr and R. T. Ng, "A unified approach for mining outliers," in *Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research.* IBM Press, 1997, p. 11.

[136] M. Wu and C. Jermaine, "Outlier detection by sampling with accuracy guarantees," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2006, pp. 767–772.

[137] H. Fan, O. Zaïane, A. Foss, and J. Wu, “A nonparametric outlier detection for effectively discovering top-n outliers from engineering data,” *Advances in Knowledge Discovery and Data Mining*, pp. 557–566, 2006.

[138] W. Jin, A. K. Tung, J. Han, and W. Wang, “Ranking outliers using symmetric neighborhood relationship.” in *PAKDD*, vol. 6. Springer, 2006, pp. 577–593.

[139] J. Tang, Z. Chen, A. Fu, and D. Cheung, “Enhancing effectiveness of outlier detections for low density patterns,” *Advances in Knowledge Discovery and Data Mining*, pp. 535–548, 2002.

[140] Z. He, S. Deng, and X. Xu, “Outlier detection integrating semantic knowledge,” in *International Conference on Web-Age Information Management.* Springer, 2002, pp. 126–131.

[141] Z. He, X. Xu, and S. Deng, “Discovering cluster-based local outliers,” *Pattern Recognition Letters*, vol. 24, no. 9, pp. 1641–1650, 2003.

[142] C. Wang, K. Viswanathan, L. Choudur, V. Talwar, W. Satterfield, and K. Schwan, “Statistical techniques for online anomaly detection in data centers,” in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on.* IEEE, 2011, pp. 385–392.

[143] P. Domingos, “Metacost: A general method for making classifiers cost-sensitive,” in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 1999, pp. 155–164.

[144] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2001.

[145] D. M. Tax and R. P. Duin, “Support vector data description,” *Machine learning*, vol. 54, no. 1, pp. 45–66, 2004.

[146] B. Zadrozny, J. Langford, and N. Abe, “Cost-sensitive learning by cost-proportionate example weighting,” in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on.* IEEE, 2003, pp. 435–442.

[147] I. Mani and I. Zhang, “knn approach to unbalanced data distributions: a case study involving information extraction,” in *Proceedings of workshop on learning from imbalanced datasets*, vol. 126, 2003.

[148] K. M. Ting, “An instance-weighting method to induce cost-sensitive trees,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 3, pp. 659–665, 2002.

[149] G. M. Weiss and F. Provost, “Learning when training data are costly: The effect of class distribution on tree induction,” *Journal of Artificial Intelligence Research*, vol. 19, pp. 315–354, 2003.

[150] M. V. Joshi, R. C. Agarwal, and V. Kumar, “Mining needle in a haystack: classifying rare classes via two-phase rule induction,” *ACM SIGMOD Record*, vol. 30, no. 2, pp. 91–102, 2001.

[151] P. Juszczak and R. P. Duin, “Uncertainty sampling methods for one-class classifiers,” in *Proceedings of ICML-03, Workshop on Learning with Imbalanced Data Sets II*, 2003, pp. 81–88.

[152] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, "Svms modeling for highly imbalanced classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 1, pp. 281–288, 2009.

[153] G. Wu and E. Y. Chang, "Class-boundary alignment for imbalanced dataset learning," in *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, 2003, pp. 49–56.

[154] D. R. Brillinger, *Time series: data analysis and theory.* Siam, 1981, vol. 36.

[155] E. S. Gardner Jr, "Exponential smoothing: The state of the art—part ii," *International journal of forecasting*, vol. 22, no. 4, pp. 637–666, 2006.

[156] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An online algorithm for segmenting time series," in *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on.* IEEE, 2001, pp. 289–296.

[157] P. Bohannon, W. Fan, M. Flaster, and R. Rastogi, "A cost-based model and effective heuristic for repairing constraints by value modification," in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data.* ACM, 2005, pp. 143–154.

[158] X. Chu, I. F. Ilyas, and P. Papotti, "Holistic data cleaning: Putting violations into context," in *2013 IEEE 29th International Conference on Data Engineering (ICDE).* IEEE, 2013, pp. 458–469.

[159] S. Song, A. Zhang, J. Wang, and P. S. Yu, "Screen: Stream data cleaning under speed constraints," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data.* ACM, 2015, pp. 827–841.

[160] A. Zhang, S. Song, J. Wang, and P. S. Yu, "Time series data cleaning: from anomaly detection to anomaly repairing," *Proceedings of the VLDB Endowment*, vol. 10, no. 10, pp. 1046–1057, 2017.

[161] G. Park, A. C. Rutherford, H. Sohn, and C. R. Farrar, "An outlier analysis framework for impedance-based structural health monitoring," *Journal of Sound and Vibration*, vol. 286, no. 1-2, pp. 229–250, 2005.

[162] H. Kang, "The prevention and handling of the missing data," *Korean journal of anesthesiology*, vol. 64, no. 5, pp. 402–406, 2013.

[163] V. Raghunathan, S. Ganeriwal, and M. Srivastava, "Emerging techniques for long lived wireless sensor networks," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 108–114, 2006.

[164] C. Alippi, G. Anastasi, C. Galperti, F. Mancini, and M. Roveri, "Adaptive sampling for energy conservation in wireless sensor networks for snow monitoring applications," in *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE International Conference on.* IEEE, 2007, pp. 1–6.

[165] M. Basseville, I. V. Nikiforov *et al.*, *Detection of abrupt changes: theory and application.* Prentice Hall Englewood Cliffs, 1993, vol. 104.

[166] R. Willett, A. Martin, and R. Nowak, "Backcasting: adaptive sampling for sensor networks," in *Proceedings of the 3rd international symposium on Information processing in sensor networks.* ACM, 2004, pp. 124–133.

[167] M. C. Vuran and I. F. Akyildiz, "Spatial correlation-based collaborative medium access control in wireless sensor networks," *IEEE/ACM Transactions On Networking*, vol. 14, no. 2, pp. 316–329, 2006.

[168] Y.-C. Tseng, Y.-C. Wang, K.-Y. Cheng, and Y.-Y. Hsieh, "imouse: an integrated mobile surveillance and wireless sensor system," *Computer*, vol. 40, no. 6, 2007.

[169] T. Kijewski-Correa, M. Haenggi, and P. Antsaklis, "Wireless sensor networks for structural health monitoring: A multi-scale approach," in *Structures Congress 2006: 17th Analysis and Computation Specialty Conference*, 2006, pp. 1–16.

[170] A. Singh, D. Budzik, W. Chen, M. A. Batalin, M. Stealey, H. Borgstrom, and W. J. Kaiser, "Multiscale sensing: a new paradigm for actuated sensing of high frequency dynamic phenomena," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on.* IEEE, 2006, pp. 328–335.

[171] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong, "Approximate data collection in sensor networks using probabilistic models," in *null.* IEEE, 2006, p. 48.

[172] A. Jain, E. Y. Chang, and Y.-F. Wang, "Adaptive stream resource management using kalman filters," in *Proceedings of the 2004 ACM SIGMOD international conference on Management of data.* ACM, 2004, pp. 11–22.

[173] B. Kanagal and A. Deshpande, "Online filtering, smoothing and probabilistic modeling of streaming data," in *2008 IEEE 24th International Conference on Data Engineering.* IEEE, 2008, pp. 1160–1169.

[174] D. Tulone and S. Madden, "Paq: Time series forecasting for approximate query answering in sensor networks," in *European Workshop on Wireless Sensor Networks.* Springer, 2006, pp. 21–37.

[175] ——, "An energy-efficient querying framework in sensor networks for detecting node similarities," in *Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems.* ACM, 2006, pp. 191–300.

[176] J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, and A. Marrs, *Disruptive technologies: Advances that will transform life, business, and the global economy.* McKinsey Global Institute San Francisco, CA, 2013, vol. 180.

[177] V. Turner, J. F. Gantz, D. Reinsel, and S. Minton, "The digital universe of opportunities: Rich data and the increasing value of the internet of things," *IDC Analyze the Future*, vol. 16, 2014.

[178] G. Fersi, "Middleware for internet of things: A study," in *Distributed Computing in Sensor Systems (DCOSS), 2015 International Conference on.* IEEE, 2015, pp. 230–235.

[179] K. Aberer, M. Hauswirth, and A. Salehi, "The global sensor networks middleware for efficient and flexible deployment and interconnection of sensor networks," Tech. Rep., 2006.

[180] S. Farrell, "Low-power wide area network (lpwan) overview," Tech. Rep., 2018.

[181] J. Petäjäjärvi, K. Mikhaylov, R. Yasmin, M. Hämäläinen, and J. Iinatti, "Evaluation of lora lpwan technology for indoor remote health and wellbeing monitoring," *International Journal of Wireless Information Networks*, vol. 24, no. 2, pp. 153–165, 2017.

[182] T. Zahariadis, A. Papadakis, F. Alvarez, J. Gonzalez, F. Lopez, F. Facca, and Y. Al-Hazmi, "Fiware lab: managing resources and services in a cloud federation supporting future internet applications," in *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on.* IEEE, 2014, pp. 792–799.

[183] "Fiware iot agent specification." [Online]. Available: http://fiware-iot-stack.readthedocs.io/en/latest/device_gateway/

[184] K. Aberer, M. Hauswirth, and A. Salehi, "Global Sensor Networks," *IEEE Communications Magazine*, no. 5005, pp. 1–14, 2006.

[185] P. Ali Sahely, "The global sensor networks project," https://github.com/LSIR/gsn, 2004.

[186] M. Eisenhauer, P. Rosengren, and P. Antolin, "A development platform for integrating wireless devices and sensors into ambient intelligence systems," in *Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2009. SECON Workshops' 09. 6th Annual IEEE Communications Society Conference on.* IEEE, 2009, pp. 1–3.

[187] K. project, "Kaa overview - kaa open-source iot platform," www.kaaproject.org/overview, 2016.

[188] M. Köhler, D. Wörner, and F. Wortmann, "Platforms for the internet of things–an analysis of existing solutions," in *5th Bosch Conference on Systems and Software Engineering (BoCSE)*, 2014.

[189] H. Derhamy, J. Eliasson, J. Delsing, and P. Priller, "A survey of commercial frameworks for the internet of things," in *IEEE International Conference on Emerging Technologies and Factory Automation: 08/09/2015-11/09/2015.* IEEE Communications Society, 2015.

[190] P. E. I. Solutions, "Platform technology: Thingworx. 2016," *URL: https://www. thingworx. com/(cited on page 25).*

[191] S. K. Datta, C. Bonnet, and N. Nikaein, "An iot gateway centric architecture to provide novel m2m services," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on.* IEEE, 2014, pp. 514–519.

[192] S. K. Datta and C. Bonnet, "Easing iot application development through datatweet framework," in *Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on.* IEEE, 2016, pp. 430–435.

[193] D. Guinard, V. Trifa, and E. Wilde, "A resource oriented architecture for the web of things," in *Internet of Things (IOT), 2010.* IEEE, 2010, pp. 1–8.

[194] I. Joyent, "Nodejs platform."

[195] A. M. Nagib and H. S. Hamza, "Sighted: a framework for semantic integration of heterogeneous sensor data on the internet of things," *Procedia Computer Science*, vol. 83, pp. 529–536, 2016.

[196] S. K. Datta and C. Bonnet, "A lightweight framework for efficient m2m device management in onem2m architecture," in *Recent Advances in Internet of Things (RIoT), 2015 International Conference on.* IEEE, 2015, pp. 1–6.

[197] M. B. Rosson and J. M. Carroll, *Usability engineering: scenario-based development of human-computer interaction.* Morgan Kaufmann, 2002.

[198] D. Floyer, *Defining and sizing the industrial Internet.* Marlborough, MA, USA: Wikibon, 2013.

[199] S. K. A. P. C. Julien, "Virtual sensors: Abstracting data from physical sensors," in *Proceedings of the 2006 International Symposium on on World of Wireless Mobile and Multimedia Networks*, 2006, pp. 587–592.

[200] A. Gupta and N. Mukherjee, "Implementation of virtual sensors for building a sensor-cloud environment," in *2016 8th International Conference on Communication Systems and Networks (COMSNETS).* Bangalore, 2016, pp. 1–8.

[201] I. L. C. E. C. M. J. Crowcroft, "Senshare: Transforming sensor networks into multi-application sensing infrastructures," in *Proceedings of the 9th European Conference on Wireless Sensor Networks*, 2012, pp. 65–81.

[202] N. S. Bose A. Gupta S. Adhikary, ""towards a sensor cloud infrastructure with sensor virtualization"," in *15,.* Proceedings of the Second Workshop on Mobile Sensing Computing and Communication ser. MSCC, 2015, pp. 25–30.

[203] F. Adelantado, X. Vilajosana, P. Tuset, B. Martinez, and J. M.-S. et al., *Understanding the Limits of LoRaWAN.* Institute of Electrical and Electronics Engineers: IEEE Communications Magazine, 2017.

[204] S. Madria, V. Kumar, and R. Dalvi, "Sensor cloud: A cloud of virtual sensors," *IEEE Software*, vol. 31, pp. 70–77, 2014.

[205] S. Bose, A. Gupta, S. Adhikary, and N. Mukherjee, "Towards a sensor-cloud infrastructure with sensor virtualization," in *15), New York, NewYork, USA,.* Computing and Communication(MSCC: Proceedings of the Second Workshop on Mobile Sensing, 2015, pp. 25–30.

[206] J. Z. al., "Supporting personizable virtual internet of things," in *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing.* Vietri sul Mere, 2013, pp. 329–336.

[207] Y. Jeong, H. Joo, G. Hong, D. Shin, and S. Lee, "Aviot: web-based interactive authoring and visualization of indoor internet of things," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 3, pp. 295–301, August 2015.

[208] X. Z. D. E. P. C. Julien, "Braceforce: a middleware to enable sensing integration in mobile applications for novice programmers," in *Proceedings ACM International Conference on Mobile Software Engineering and Systems*, June 2014, pp. 8–17.

[209] K. L. Hu F. Wang J. Zhou, ""a data processing middleware based on soa for the internet of things"," *Journal of Sensor*, January 2015.

[210] D. Brunelli, G. Gallo, and L. Benini, *Sensormind: Virtual Sensing and Complex Event Detection for Internet of Things.* Environment and Society. ApplePies Lecture Notes in Electrical Engineering, Springer, Cham: De Gloria A. Applications in Electronics Pervading Industry, 2016, vol. 409.

[211] Z. Shelby and I. R. 6690:, *Constrained RESTful Environments (CoRE) Link Format.* IETF, 2012.

[212] I. Joyent, "About node. js," 2014.

[213] A. CouchDB, "Apache couchdb."

[214] S. K. Datta, A. Gyrard, C. Bonnet, and K. Boudaoud, "onem2m architecture based user centric iot application development," in *2015 3rd International Conference on Future Internet of Things and Cloud.* Rome, 2015, pp. 100–107.

[215] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," *IDC iView: IDC Analyze the future*, vol. 2007, no. 2012, pp. 1–16, 2012.

[216] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper*, vol. 1, no. 2011, pp. 1–11, 2011.

[217] S. Mumtaz, A. Alsohaily, Z. Pang, A. Rayes, K. F. Tsang, and J. Rodriguez, "Massive internet of things for industrial applications: Addressing wireless iiot connectivity challenges and ecosystem fragmentation," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 28–33, 2017.

[218] S. Taylor, "The next generation of the internet revolutionizing the way we work, live, play, and learn," *CISCO, San Francisco, CA, USA, CISCO Point of View*, vol. 12, 2013.

[219] NorthStream, "White paper: Massive iot: different technologies for different needs," Tech. Rep., 2017.

[220] R. S. Sinha, Y. Wei, and S.-H. Hwang, "A survey on lpwa technology: Lora and nb-iot," *Ict Express*, vol. 3, no. 1, pp. 14–21, 2017.

[221] C. Chen and A. Helal, "Device integration in soda using the device description language," in *Applications and the Internet, 2009. SAINT'09. Ninth Annual International Symposium on.* IEEE, 2009, pp. 100–106.

[222] A. E. Khaled, A. Helal, W. Lindquist, and C. Lee, "Iot-ddl–device description language for the "t" in iot," *IEEE Access*, vol. 6, pp. 24 048–24 063, 2018.

[223] "Experimental Implementation of the Web of Things Framework," https://github.com/w3c/web-of-things-framework, 2016.

[224] S. K. Datta and C. Bonnet, "Describing things in the internet of things: From core link format to semantic based descriptions," in *2016 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, May 2016, pp. 1–2.

[225] Y. Xu and A. Helal, "Scalable cloud-sensor architecture for the internet of things." *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 285–298, 2016.

[226] R. Bose, J. King, H. El-Zabadani, S. Pickles, and A. Helal, "Building plug-and-play smart homes using the atlas platform," in *Proceedings of the 4th International Conference on Smart Homes and Health Telematic (ICOST), Belfast, the Northern Islands.* Citeseer, 2006, pp. 265–272.

[227] C. Chen, R. Bose, and A. Helal, "Atlas: An open model for automatic integration and tele-programming of smart objects," in *Proceedings of the 3rd International Workshop on Design and Integration Principles for Smart Objects (DIPSO 2009), In conjunction with Ubicomp*, 2009.

[228] "W3C Web of Things (WoT) Thing Description," https://github.com/w3c/wot-thing-description.

[229] S. Kaebisch and D. Anicic, "Thing description as enabler of semantic interoperability on the web of things," in *Proc. IoT Semantic Interoperability Workshop*, 2016, pp. 1–3.

[230] D. Guinard, V. Trifa, T. Pham, and O. Liechti, "Towards physical mashups in the web of things," in *Networked Sensing Systems (INSS), 2009 Sixth International Conference on.* IEEE, 2009, pp. 1–4.

[231] S. Mayer, D. Guinard, and V. Trifa, "Facilitating the integration and interaction of real-world services for the web of things," *Proceedings of Urban Internet of Things–Towards Programmable Real-time Cities (UrbanIOT)*, 2010.

[232] B. Ostermaier, F. Schlup, and K. Römer, "Webplug: A framework for the web of things," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on.* IEEE, 2010, pp. 690–695.

[233] B. Christophe, M. Boussard, M. Lu, A. Pastor, and V. Toubiana, "The web of things vision: Things as a service and interaction patterns," *Bell labs technical journal*, vol. 16, no. 1, pp. 55–61, 2011.

[234] A. Orestis, A. Dimitrios, and C. Ioannis, "Towards integrating iot devices with the web," in *Emerging Technologies & Factory Automation (ETFA), 2012 IEEE 17th Conference on.* IEEE, 2012, pp. 1–4.

[235] A. Moraru, D. Mladenic, M. Vucnik, M. Porcius, C. Fortuna, and M. Mohorcic, "Exposing real world information for the web of things," in *Proceedings of the 8th International Workshop on Information Integration on the Web: in conjunction with WWW 2011.* ACM, 2011, p. 6.

[236] xively by LogMeln. (2014) Public Cloud for the Internet of Things. [Online]. Available: https://xively.com

[237] MathWorks. (2014) Internet of Things—ThingSpeak. [Online]. Available: https://www.thingspeak.com

[238] PTC. (2014) ThingWorx the 1st application platform for the connected world. [Online]. Available: http://www.thingworx.com/

[239] C. Y. Chen, J. H. Fu, T. Sung, P.-F. Wang, E. Jou, and M.-W. Feng, "Complex event processing for the internet of things and its applications," in *Automation Science and Engineering (CASE), 2014 IEEE International Conference on.* IEEE, 2014, pp. 1144–1149.

[240] S. I. Goldberg, A. Niemierko, and A. Turchin, "Analysis of data errors in clinical research databases," in *AMIA annual symposium proceedings*, vol. 2008. American Medical Informatics Association, 2008, p. 242.

[241] sportless. (2017) The importance of cleaning data. [Online]. Available: https://web.archive.org/web/20171205042031/https://spotlessdata.com/blog/importance-data-cleaning-user-generated-content

[242] A. A. Alkhatib, "A review on forest fire detection techniques," *International Journal of Distributed Sensor Networks*, vol. 10, no. 3, p. 597368, 2014.

[243] K. G. Mehrotra, C. K. Mohan, and H. Huang, *Anomaly Detection Principles and Algorithms.* Springer, 2017.

[244] A. Stanway, "Etsy skyline," https://github.com/etsy/skyline, 2013.

[245] T. Pevný, "Loda: Lightweight on-line detector of anomalies," *Machine Learning*, vol. 102, no. 2, pp. 275–304, Feb 2016. [Online]. Available: https://doi.org/10.1007/s10994-015-5521-0

[246] E. Burnaev and V. Ishimtsev, "Conformalized density-and distance-based anomaly detection in time-series data," *arXiv preprint arXiv:1608.04585*, 2016.

[247] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.

[248] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Procedia Computer Science*, vol. 60, pp. 708–713, 2015.

[249] A. M. Gonbadi, S. H. Tabatabaei, and E. J. M. Carranza, "Supervised geochemical anomaly detection by pattern recognition," *Journal of Geochemical Exploration*, vol. 157, pp. 81–91, 2015.

[250] J. Ma, L. Sun, H. Wang, Y. Zhang, and U. Aickelin, "Supervised anomaly detection in uncertain pseudoperiodic data streams," *ACM Transactions on Internet Technology (TOIT)*, vol. 16, no. 1, p. 4, 2016.

[251] J. Huang, Q. Zhu, L. Yang, and J. Feng, "A non-parameter outlier detection algorithm based on Natural Neighbor," *Knowledge-Based Systems*, vol. 92, pp. 71–77, jan 2016. [Online]. Available: http://dx.doi.org/10.1016/j.knosys.2015.10.014http://linkinghub.elsevier.com/retrieve/pii/S0950705115004013

[252] X. Luo, Y. Xia, Q. Zhu, and Y. Li, "Boosting the k-nearest-neighborhood based incremental collaborative filtering," *Knowledge-Based Systems*, vol. 53, pp. 90–99, 2013.

[253] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[254] P.-E. Danielsson, "Euclidean distance mapping," *Computer Graphics and image processing*, vol. 14, no. 3, pp. 227–248, 1980.

[255] B. Andrews, "Classification of limiting shapes for isotropic curve flows," *Journal of the American mathematical society*, vol. 16, no. 2, pp. 443–459, 2003.

[256] "Welcome to the sax," http://www.cs.ucr.edu/~eamonn/SAX.htm, (Accessed on 02/05/2019).

[257] E. Keogh, J. Lin, and A. Fu, "Hot sax: Efficiently finding the most unusual time series subsequence," in *null.* Ieee, 2005, pp. 226–233.

[258] X. Xi, E. Keogh, L. Wei, and A. Mafra-Neto, "Finding motifs in a database of shapes," in *Proceedings of the 2007 SIAM international conference on data mining.* SIAM, 2007, pp. 249–260.

[259] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery.* ACM, 2003, pp. 2–11.

[260] R. Killick and I. Eckley, "changepoint: An r package for changepoint analysis," *Journal of Statistical Software*, vol. 58, no. 3, pp. 1–19, 2014.

[261] S. Liu, M. Yamada, N. Collier, and M. Sugiyama, "Change-point detection in time-series data by relative density-ratio estimation," *Neural Networks*, vol. 43, pp. 72–83, 2013.

[262] Y. Kawahara and M. Sugiyama, "Change-point detection in time-series data by direct density-ratio estimation," in *Proceedings of the 2009 SIAM International Conference on Data Mining.* SIAM, 2009, pp. 389–400.

[263] V. Guralnik and J. Srivastava, "Event detection from time series data," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 1999, pp. 33–42.

[264] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining.* IEEE, 2008, pp. 413–422.

[265] J. Ma and S. Perkins, "Time-series novelty detection using one-class support vector machines," in *Neural Networks, 2003. Proceedings of the International Joint Conference on*, vol. 3. IEEE, 2003, pp. 1741–1745.

[266] P. J. Rousseeuw and K. V. Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, no. 3, pp. 212–223, 1999.

[267] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration." *VISAPP (1)*, vol. 2, no. 331-340, p. 2, 2009.

[268] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and information systems*, vol. 51, no. 2, pp. 339–367, 2017.

[269] J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic anomaly detection in the cloud via statistical learning," *arXiv preprint arXiv:1704.07706*, 2017.

[270] V. Carey, E. Walters, C. Wager, and B. Rosner, "Resistant and test-based outlier rejection: effects on gaussian one-and two-sample inference," *Technometrics*, vol. 39, no. 3, pp. 320–330, 1997.

[271] A. Liaw, M. Wiener *et al.*, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[272] M. Kuss and C. E. Rasmussen, “Assessing approximate inference for binary gaussian process classification,” *Journal of machine learning research*, vol. 6, no. Oct, pp. 1679–1704, 2005.

[273] P. Viola and M. Jones, “Fast and robust classification using asymmetric adaboost and a detector cascade,” in *Advances in neural information processing systems*, 2002, pp. 1311–1318.

[274] D. Cohn, L. Atlas, and R. Ladner, “Improving generalization with active learning,” *Machine learning*, vol. 15, no. 2, pp. 201–221, 1994.

[275] J. He, E. Veltri, D. Santoro, G. Li, G. Mecca, P. Papotti, and N. Tang, “Interactive and deterministic data cleaning,” in *Proceedings of the 2016 International Conference on Management of Data*. ACM, 2016, pp. 893–907.

[276] smirmik. (2016) Contextual anomaly detector. [Online]. Available: https://github.com/smirmik/CAD

[277] R. P. Adams and D. J. MacKay, “Bayesian online changepoint detection,” *arXiv preprint arXiv:0710.3742*, 2007.

[278] H.-P. Kriegel, A. Zimek *et al.*, “Angle-based outlier detection in high-dimensional data,” in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 444–452.

[279] A. Lazarevic and V. Kumar, “Feature bagging for outlier detection,” in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 157–166.

[280] J. Hardin and D. M. Rocke, “Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator,” *Computational Statistics & Data Analysis*, vol. 44, no. 4, pp. 625–638, 2004.

[281] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, “A novel anomaly detection scheme based on principal component classifier,” MIAMI UNIV CORAL GABLES FL DEPT OF ELECTRICAL AND COMPUTER ENGINEERING, Tech. Rep., 2003.

[282] a. yuezhao, “Python outlier detection (pyod),” https://github.com/yzhao062/pyod, 2017.

[283] J. Ha, S. Seok, and J.-S. Lee, “Robust outlier detection using the instability factor,” *Knowledge-Based Systems*, vol. 63, pp. 15–23, 2014.

[284] J. Huang, Q. Zhu, L. Yang, and J. Feng, “A non-parameter outlier detection algorithm based on natural neighbor,” *Knowledge-Based Systems*, vol. 92, pp. 71–77, 2016.

[285] A. Lakhina, M. Crovella, and C. Diot, “Diagnosing network-wide traffic anomalies,” *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 219–230, Aug. 2004. [Online]. Available: http://doi.acm.org/10.1145/1030194.1015492

[286] E. Keogh, J. Lin, and A. Fu, “Hot sax: efficiently finding the most unusual time series subsequence,” in *Fifth IEEE International Conference on Data Mining (ICDM’05)*, Nov 2005, pp. 8 pp.–.

[287] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[288] C. M. Sadler and M. Martonosi, "Data compression algorithms for energy-constrained devices in delay tolerant networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems.* ACM, 2006, pp. 265–278.

[289] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: A tiny aggregation service for ad-hoc sensor networks," *ACM SIGOPS Operating Systems Review*, vol. 36, no. SI, pp. 131–146, 2002.

[290] A. Boulis, S. Ganeriwal, and M. B. Srivastava, "Aggregation in sensor networks: an energy–accuracy trade-off," *Ad hoc networks*, vol. 1, no. 2-3, pp. 317–331, 2003.

[291] S. Goel and T. Imielinski, "Prediction-based monitoring in sensor networks: taking lessons from mpeg," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 5, pp. 82–98, 2001.

[292] P. Ellis, "Extension of phase plane analysis to quantized systems," *IRE Transactions on Automatic Control*, vol. 4, no. 2, pp. 43–54, 1959.

[293] N. Persson and F. Gustafsson, "Event based sampling with application to vibration analysis in pneumatic tires," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, vol. 6, 2001, pp. 3885–3888 vol.6.

[294] ——, "Event based sampling with application to vibration analysis in pneumatic tires," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 6. IEEE, 2001, pp. 3885–3888.

[295] P. G. Otanez, J. R. Moyne, and D. M. Tilbury, "Using deadbands to reduce communication in networked control systems," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 4. IEEE, 2002, pp. 3015–3020.

[296] M. Miskowicz, "Sampling of signals in energy domain," in *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on*, vol. 1. IEEE, 2005, pp. 4–pp.

[297] Y. S. Suh, "Send-on-delta sensor data transmission with a linear predictor," *Sensors*, vol. 7, no. 4, pp. 537–547, 2007.

[298] J. Ploennigs, V. Vasyutynskyy, and K. Kabitzsch, "Comparison of energy-efficient sampling methods for wsns in building automation scenarios," in *Emerging Technologies & Factory Automation, 2009. ETFA 2009. IEEE Conference on.* IEEE, 2009, pp. 1–8.

[299] T. Shu, M. Xia, J. Chen, and C. de Silva, "An energy efficient adaptive sampling algorithm in a sensor network for automated water quality monitoring," *Sensors*, vol. 17, no. 11, p. 2551, 2017.

[300] T. Blu, P. Thévenaz, and M. Unser, "Linear interpolation revitalized," *IEEE Transactions on Image Processing*, vol. 13, no. 5, pp. 710–719, 2004.

[301] Y. Sun, H. Song, A. J. Jara, and R. Bie, "Internet of things and big data analytics for smart and connected communities," *IEEE access*, vol. 4, pp. 766–773, 2016.