MASTER OF SCIENCE
IN ENGINEERING

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts
Western Switzerland

Master of Science HES-SO in Engineering
Av. de Provence 6
CH-1007 Lausanne

Master of Science HES-SO in Engineering

Orientation: Information and Communication Technologies (ICT)

Setting up and deploying a machine learning based edge-to-cloud platform on a power microgrid infrastructure

Author:
Raoul Dupuis

Under the direction of:
Prof. Nabil Abdennadher
inIT

External expert:
Pascal Kienast

Lausanne, HES-SO//Master, 2022

Information about this report

Contact information

Author: Raoul Dupuis
MSE Student
HES-SO//Master
Switzerland
Email: [raoul.dupuis@master.hes-so.ch]

Declaration of honor

I, undersigned, Raoul Dupuis, hereby declare that the work submitted is the result of a personal work. I certify that I have not resorted to plagiarism or other forms of fraud. All sources of information used and the author quotes were clearly mentioned.

Place, date: ______________________________

Signature: ______________________________

Validation

Accepted by the HES-SO//Master (Switzerland, Lausanne) on a proposal from:

Prof. Nabil Abdennadher, Thesis project advisor

Pascal Kienast, Clemap, Main expert

Place, date: ______________________________

Prof. Nabil Abdennadher

Advisor

Contents

List of Figures

List of Tables

1 | Introduction

1.1 Context

The need for implementing an energy transition in favour of renewable and clean energy has become urgent and obvious. Many governments mention this issue as one of their top priorities and numerous programs, such as the Swiss "Energy Strategy 2050" [1], have been developed to maintain high standards of supply based on renewable energy sources while minimizing environmental damage.

Besides, new forms of electricity production and consumption are disrupting the market. Renewable energy sources (solar and wind) are, by definition, intermittent and distributed in vast numbers of small-scale decentralized energy producers, which affects grid management fundamentally. The decentralized production of electricity indeed permits self-consumption of electricity (prosumer) and creation of peer-to-peer networks: microgrids. Microgrids offer the regulatory framework and the economic incentive to support decentralised renewable energy sources and provide neighbourhoods with solutions to actively engage in energy economy.

In parallel, consumption scenarios are changing as fossil fuels are being replaced by electricity in a series of domains (electric vehicles, batteries in buildings, etc.). New digital technologies can enable this energy transformation. The development of digital meters (commonly called "smart meters") is only one step in this transformation. But these digital meters, in their present form will not be able to cope with the various challenges to be met. They are not yet ready for a resilient and future-oriented grid configuration due to low processing resources, no remote update functionality, data integrity and connectivity issues. Originally designed to remotely measure and monitor the overall electricity consumption of households, the capabilities of these digital meters will be extended to perform smart purposes such as:

- Monitoring, controlling, and exchanging information between home appliances to optimize local or "microgrid" consumption.
- Communication: gather information about other households.
- Predicting local (households) and global (microgrid) consumption/production.
- Setting-up power transactions, negotiation with other households/microgrids.

These new functionalities will be embedded in the digital meters or integrated through a dedicated hardware that will be connected to the digital meter: Grid Edge Device (GED). Intelligent microgrids (microgrids empowered with GEDs) would be based on AI mechanisms capable of forecasting power usage/production. These algorithms will be the foundation to build end-user context-aware and self-adaptive energy applications such as power network stability services and power transactions/flexibility negotiations. From our perspective, an intelligent micro-grid is powered by a digital platform composed of a hardware infrastructure and AI based tools:

- The hardware infrastructure is composed of three layers: home appliances (heat pumps, boilers, EV charging stations, etc.), grid edge device (GED), and cloud infrastructure.
- The AI based tools are composed of a set of collaborative ML based algorithms deployed on the GEDs and the Cloud. The main aim of AI based tools is to ensure self-adaptive management of micro-grids. The platform is composed of a set of couples <Learning, Prediction> modules. Each couple of modules is responsible for managing a particular aspect of the micro-grid: predicting local/global consumption/production, predicting peaks of consumption/prediction, etc.

From an IT perspective, Intelligent micro-grids rely on edge-to-cloud solutions where the edge devices are monitored and controlled by the cloud.The objective of this master's work is to install an intelligent micro-grid at the Meyrin school. For this purpose we will rely on the following technologies:

- CLEMAP devices will be used as GEDs.
- Nuvla/NuvlaBox technology will be our basic building blocs to setup an efficient edge-to-cloud framework.
- A set of learning algorithms used to generate Machine Learning Based modules to forecast prediction/consumption at the level of households and micro-grids.

1.2 Report plan

This document is organized as follow:
chapter 2 gives an overview of some smart grid solutions being proposed in industry and academia. These solutions are often based on Cloud or Edge-To-Cloud architectures.The chapter focuses on the GEDs used in these smart grid solutions.

chapter 3 provides an overview of the edge-to-cloud solutions available on the market. These solutions could be used as cornerstone to build an edge-to-cloud platform for intelligent microgrids
chapter 4 will deal with the deployment of an intelligent microgrid in Meyrin school.
chapter 5 provides an overview of the state of the art of Machine Learning (ML) algorithms applied to energy consumption/production forecast.
Finally **chapter 6** will detail the ML algorithm applied on power consumption/production data from the perspective of being deployed on Meyrin microgrid.

2 | Digital platforms for smart micro-grids

2.1 Context

The goal of this chapter is to review the state-of-the-art of the digital platforms used to develop and deploy smart micro-grids. Before going into deeper detail, it is worth defining what micro-grid is.

Renewable energy sources (solar and wind) are by definition intermittent, which affects grid management fundamentally. This energy is produced either in a power plant (E.g. large scale wind farms) or distributed over many small-scale decentralized producers (E.g. solar panels on the roof of a house). The decentralized production of electricity permits self-consumption of electricity (prosumer, see Figure 2.1) and creation of peer-to-peer distribution networks (micro-grids, see Figure 2.2). A microgrid can be defined according to the "IEEE Standard for the Specification of Microgrid Controllers" [2] in the following way:

- A microgrid is a group of interconnected loads and distributed energy resources within clearly defined electrical boundaries that acts as a single controllable entity with respect to the grid.
- A microgrid can connect and disconnect from the grid to enable it to operate in both grid-connected or island mode.

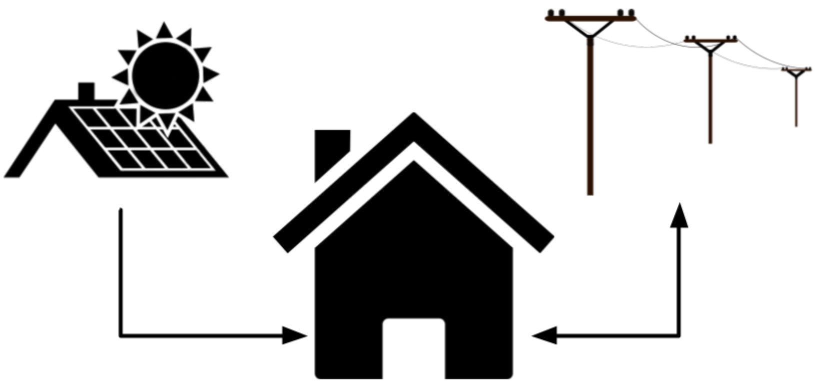

Figure 2.1 Prosumer: households that are consumers and/or producers

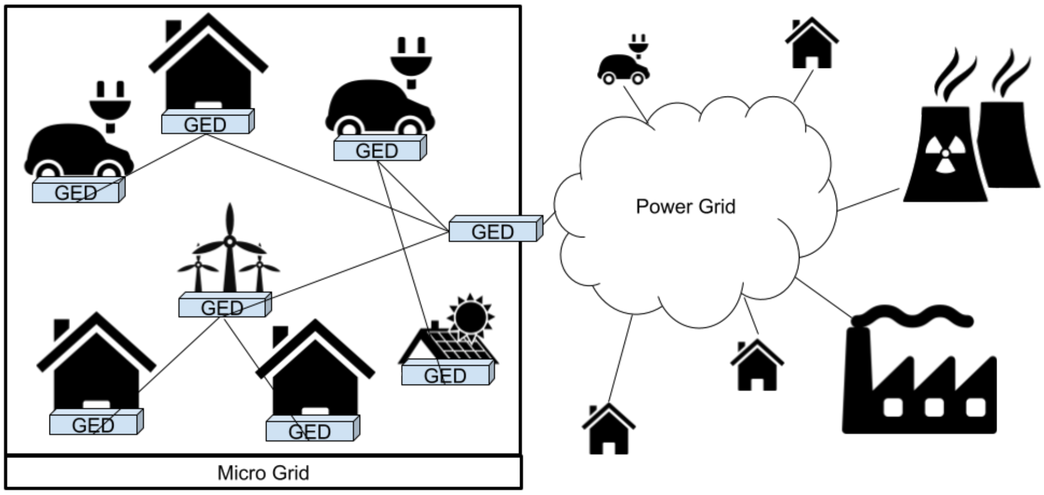

Figure 2.2 Micro grid connected to power grid

Digital platforms represented by the bottom dotted rectangle in the Figure 2.3 will be the building blocks to develop end-user self-adaptive application (top dotted rectangle) for smart grid. These digital platforms deployed on top of micro-grids are often composed of three layers:

1. The infrastructure that represents the copper electrical network. All appliances that consume, produce and store energy (such as heat pump, boilers, EV charging station, PV panels ...) are also included in this layer.

2. A digital infrastructure composed of grid edge device (GED) and cloud infrastructure with edge-to-cloud environment.

3. A set of basic software services deployed on the GEDs and/or the cloud. These services can be of different types according to the platform: data visualization, device controls, predicting consumption and/or production, signature recognition (to identify used devices), etc. These services will be used as cornerstone to develop high level end-user applications

The intelligence is shared between the edge and the cloud. In some digital platforms, this intelligence is fully implemented in the cloud. The GEDs just act as a relay. In other platforms, the intelligence is deployed in the GED and can be adapted/improved by the cloud, locally with new measurement, or via a peer-to-peer collaboration.

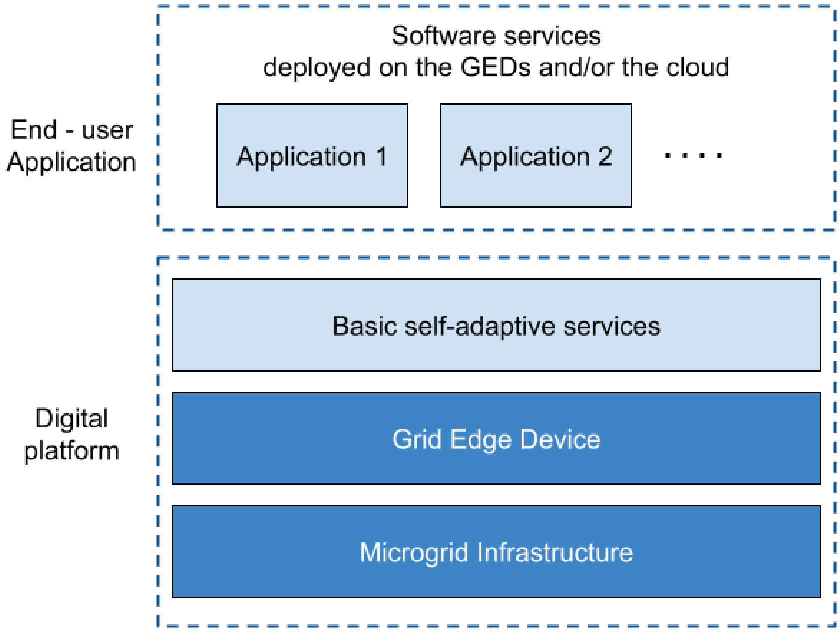

Figure 2.3 End-user self-adaptive application deployed on digital platform

The goal of this chapter is to compare four digital platforms with respect to the following criteria:

- **P2P trading** Peer to peer trading between neighbors in the same micro-grid. This is an important feature in a micro-grid, so that the household that produces more electricity than it needs can sell it to its neighbor. Block chain technology can be used to secure transactions.
- **Load disaggregation** at home device level. The aim is to identify the appliances that are consuming and/or producing power. Here, two alternatives are possible. The first is to install sensors at each appliance to measure its consumption and/or production. These measurements are then forwarded to the GED or the cloud. A second alternative is to deduce this information from the global consumption/production measurement of the household.
- **Forecast** In order to anticipate transactions and optimize energy use, the digital platform must predict future consumption and production.
- **Flexibility** From the perspective of balancing the grid, and use renewable energy in a more efficient way, the consumer can trade flexibility. An intelligent device activates/deactivates at the appropriate time consumer and storage devices according to the production and consumption forecasts.

2.2 SMA

SMA [3] is a global German company that offers a complete management solution for owners of solar PhotoVoltaics (PV) panels with hardware and software products. They have equipment for private households, but also for large-scale green energy production.

A SMA basic configuration uses the solar production only for local consumption. The surplus is lost. With additional smart devices, an SMA installation can manage over-productions and provide load balancing.
With the addition of a connected electrical outlet or with some compatible devices that uses EEBUS protocol [4], the system can monitor the usage consumption of each device and activate them at the appropriate time.

The main power grid is not designed to accept more than 70% of the peak power of solar production [5]. Instead of simply limited the inverter of the solar panel, the GED of SMA (sunny home manager 2.0 [6]) allows to regulate this too important production thanks to the forecast of power production and the use of battery and flexible devices.

Figure 2.4 shows a SMA installation in a household with a solar production, home appliances and a battery. The sunny portal supervises the system.

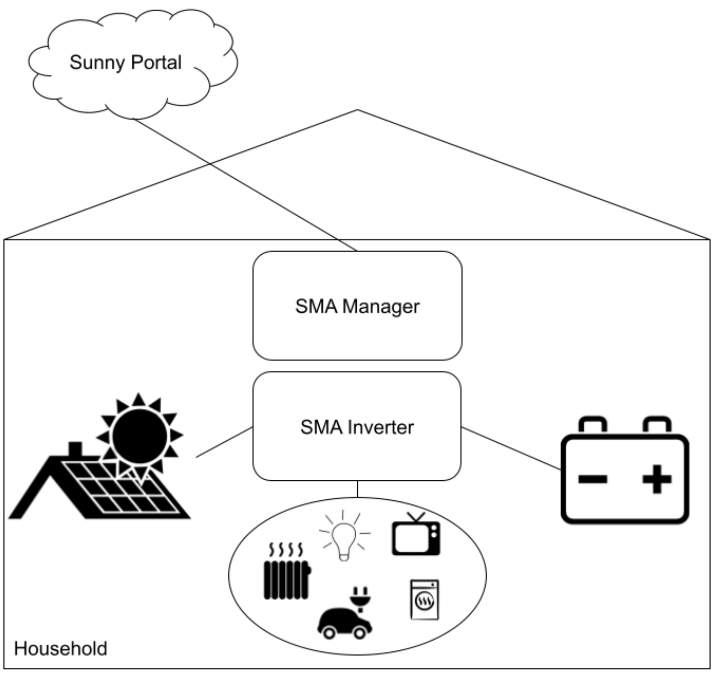

Figure 2.4 SMA installation diagram

2.3 Hive Power

Founded in 2017 in Switzerland, Hive Power [7] is a leading provider of innovative solutions for smart grids. They develop software solutions for micro-grid management, with features such as price management or anomaly detection

The Figure 2.5 shows the installation of a micro-grid named Lugaggia Innovation Community (LIC) [8], in a village in Ticino. This project was initiated to use the solar energy produced by the kindergarten, which is not consume during the summer vacations. Several consumers are connected in this micro-grid. The energy produced is stored in batteries, and consumed in the neighborhood. They use two types of GED described below.

- **Strato Pi** [9] Is an edge device based on Raspberry Pi Compute Module. It is connected to the Landis+Gyr E450 which is a metering device widely used for electricity smart metering. The communication between the two devices is done by optical serial interface.
- **NUC** [10] Is a much more powerful and robust edge device, provided by Intel. It acts like an aggregator in the community running specific control applications that cannot be managed by the Strato Pi devices.

The Strato Pi brings GED capabilities to the smart meter. The NUC is a more powerful GED connected near a battery, performing tasks requiring more resources.

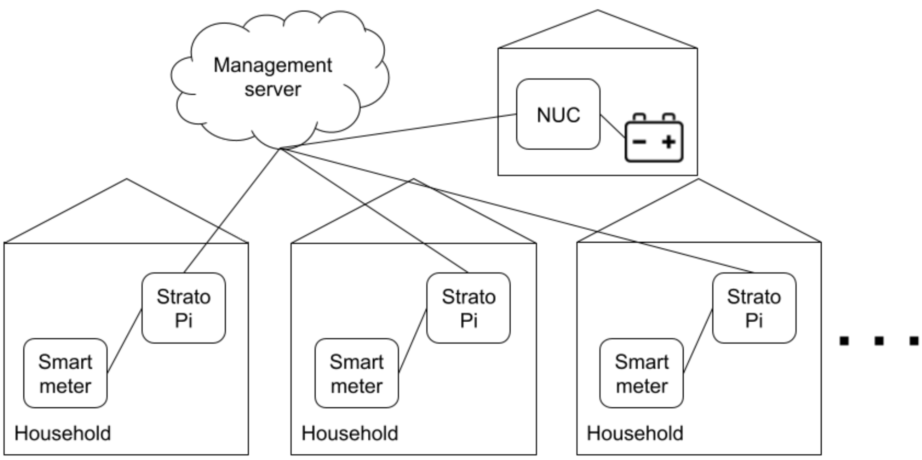

Figure 2.5 LIC installation diagram

To develop their activities, hive power has participated in several research projects. The first one that allowed the creation of hive power is NEMoGrid [11][12]. This project funded by ERA-Net Smart Energy Systems, is intended to favouring the grid-integration of decentralized energy resources. Three pilot projects across Europe (Switzerland, Sweden and Germany) have allowed testing the models.

The EU-funded PARITY [13] project aim to stabilize the grid disturbed by the production of prosumers, which injects their production in an irregular way. For this, they used smart contracts based on block chain technology. This solution increases grid durability and efficiency, with the use of flexible devices and storage systems (battery, EVs...)

The ongoing EU-funded MAESHA [14] project develops smart and flexible methods of storage and energy management. The goal is to make a given island self-sufficient by replacing fossil fuel energy with renewable energy. In this case, the micro-grid is totally independent and disconnected from the main network.

2.4 CLEMAP

CLEMAP [15] is a Swiss company that develops a combination of hardware and software product to monitor and classify energy consumption.
They provide solutions for smart grids, smart buildings and smart factories.

They have four different type of GED named **One**, **Energy Monitor**, **Grid** and **Load Management**. They are differentiated by maximum currents supported and some additional features such as Modbus [16] interface. This Modbus interface allows sending commands for dynamic load control. The values measured by the devices can be visualized in two different interfaces depending on the product. The device is also able to classify locally usage of home appliance (freezers, dishwasher, heat pump ...) only by measuring the overall consumption.

The **App** interface available on browser or smart phone app allows visualizing the instantaneous value or grouped by day, week or month. The category of the consumer obtained by a machine learning algorithm is also displayed.
The **Floem** interface allows the client with several clemap devices to group them and have combined statistics.

The device consists of a Raspberry Pi [17] for calculation and communication, and a board allowing the measurement of voltage and current in three phases. clemap devices will be used as GED in our deployment. section 4.3 will provide more details about this device.

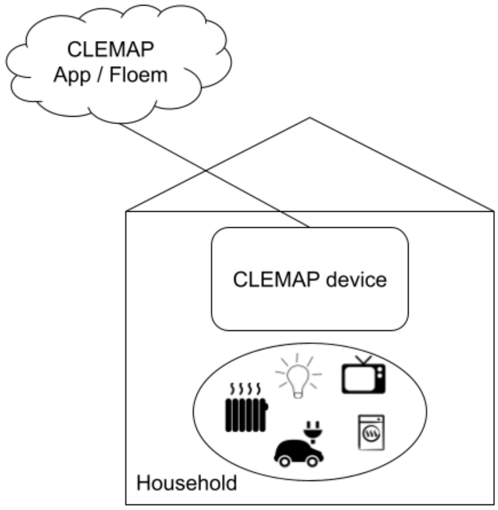

Figure 2.6 CLEMAP installation diagram

2.5 Quartierstrom

This research project [18] focuses on the exchange of electricity within a micro grid. For this, they use a GED named Smart-Pi [19]. It is composed of Raspberry Pi [17] with a board allowing connecting the voltage and current sensors. These devices are connected to a VPN in order to have a fixed IP address to facilitate Peer-to-Peer (P2P) communication. This GED allows generating smart-contracts related to the exchange of energy between neighbors, with block chain technology.

This project is not intended to classify usage to be monitored. It also does not use forecasting to act on flexible devices.

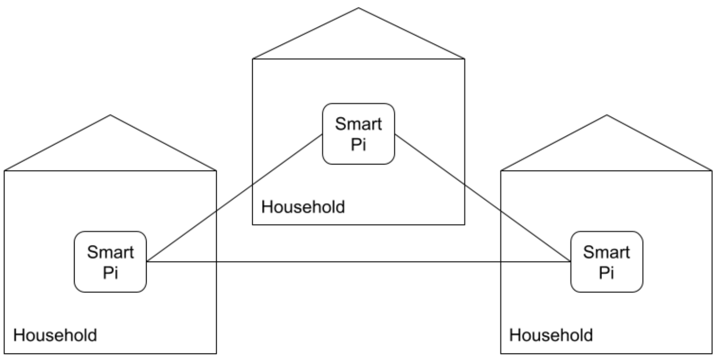

Figure 2.7 Quartierstrom installation diagram

2.6 Revolution Pi

Unlike the previous sections focus on a solution, this section is dedicated to one brand providing devices for Industrial Internet of Things (IIoT).

The Revolution Pi [20] is an Industrial PC based on a Raspberry Pi Compute Module. It is designed to be installed on a DIN rail. Thanks to an optional extension, it has many digital and analog inputs/outputs to communicate with any kind of device.

This type of device must remain power on all the time, but it can happen that the program crashes or that the power supply is cut. For this reason, this device has a hardware watchdog that allows monitoring the execution and resetting if necessary. A shutdown signals of a UPS can be received on a 24V input. It is a security system that provides power in the event of a power failure, the time to either turn off the device properly, or to receive electricity from an emergency generator. In addition, to always know what time it is, it has a Real Time Clock (RTC) with 24h buffering.

The Revolution Pi OS custom kernel based on Raspbian [21] is optimized for a real-time system, with the necessary modules to control the peripherals. The software is open source allowing it to be best adapted to its application.

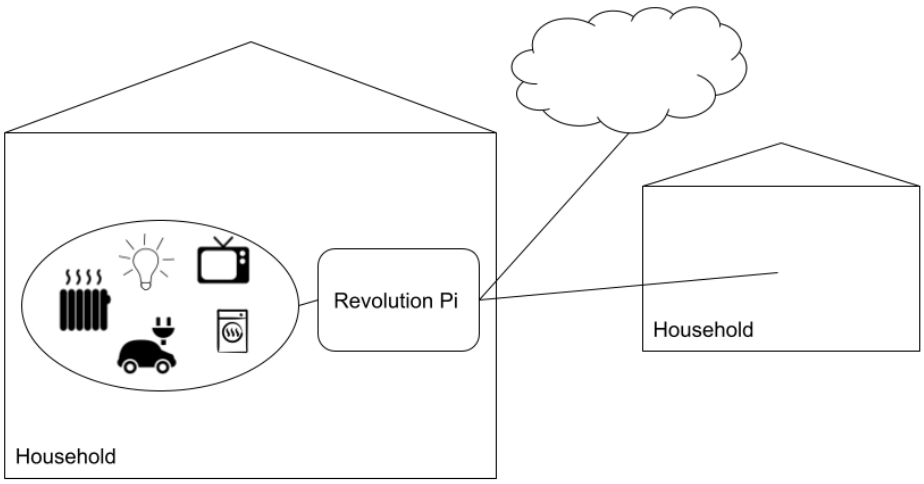

Figure 2.8 Revolution Pi diagram

2.7 Comparison summary

To summarize, all the platform described above are represented in the Table 2.1. Revolution Pi is not included in this table because is just a device.

Platform offering	P2P trading	Load disaggregation	Forecast	Flexibility
SMA	No	Yes	Yes	Yes
HIVE POWER	Yes	No	Yes	Yes
CLEMAP	No	Yes	No	Yes
Quartierstrom	Yes	No	No	No

Table 2.1 Micro-grid solution comparison

2.8 Conclusion

This chapter has focused on the study of four digital platforms used to deploy smart micro-grid. We were particularly interested in the GEDs used by these solutions and the functionalities they support: consumption/production classification, prediction and negotiation.

The solutions detailed in this chapter support integrated functions but do not support interoperability. This is an important factor to promote smart micro-grids.

Moreover, our objective is to ensure that the "intelligence" is as close as possible to the data collected from the home appliances, and therefore deployed in the GEDs.

For these two reasons (interoperability and local intelligence), the next chapter will detail candidate edge-to-cloud solutions that can deploy GEDs in an open digital ecosystem.

3 | Edge to Cloud solutions

3.1 Context

Edge computing is predicted to be a "perfect ally" for the cloud in the coming years. Pushed by the fast evolution of smart cities, industry 4.0 and self-driving cars, the need to make the edge smarter, more secure, and capable is inescapable. Self-adapting edge devices are essential to build the next generation of IoT applications that address upcoming market needs in the digital evolution. Combining the edge approach with IoT sensors and Cloud would add flexibility and choices for users.

This master's thesis falls within this scope. The main objective is to develop a smart Grid Edge Device supporting ML based functionalities. The data will then be processed (e.g., forecast, classification) locally in the GED. The traffic between the GEDs and the cloud is limited to an information exchange such as forecast results, error notifications, new improved version of the machine learning model (MLM) deployed on the GED, etc.

The performance of the MLM deployed on the GED will be continuously monitored: each time the MLM is unable to take a reliable decision, it sends the data it failed to process to the cloud. The cloud then collects these datasets from different GEDs and triggers a new learning cycle, resulting in a new intelligence (MLM) which will then be pushed back to the GED.

The aim of this chapter is to present the most well-known edge-to-cloud solutions on the market namely AWS Greengrass, Azure IoT Edge, GCP and Balena. Finally, Nuvla/NuvlaBox which is a solution developed by SixSq is also compared. Other existing solutions are listed in the last section.

The scenario illustrated in Figure 3.1 will allow us to describe and compare each solution. This is a high-level application that consumes data coming from some IoT sensors, such as sound meters, weather station, cameras, power consumption etc. A machine learning model is deployed on this edge device, which is responsible for making "predictions" on the input (sensing) data: f.i., classify environmental sounds, recognize car license plates, forecast power usage, etc. This machine learning model (MLM) is built and trained in the Cloud, where an artificial intelligence (AI) learning module is deployed; a human-supervised labeling service is also needed to prepare the training data. Generally speaking, a self-adaptive Machine Learning (ML) based application deployed on an edge-to-cloud solution is composed of a set of:

- cloud modules such as a database, learning algorithm, etc.

- edge modules endowed with MLM and optimized for a limited resource edge device.

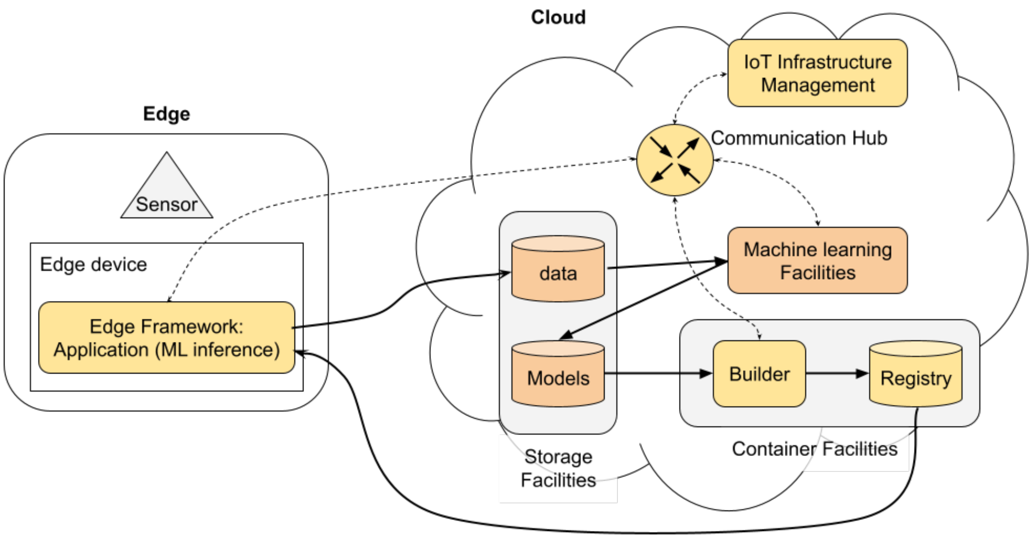

Figure 3.1 A self-adaptive ML-based edge-cloud application scenario

Each colored component in Figure 3.1 represents a module of the architecture that will be compared for each provider. In orange the components specific to the learning of a machine learning model, and in yellow the components constituting the edge to cloud infrastructure. Components are detailed below.

- **IoT Infrastructure Management** compose, provision (configure and deploy) and monitor Edge/IoT networks. That is, an orchestration service. Edge devices and their companion IoT devices are mostly autonomous; their edge framework modules are packaged into some form and provisioned to them via a Cloud repository. Thus, a bootstrapping phase is needed to prepare new edge devices with the needed Edge Framework modules.

- **Edge Framework** enable edge modules programming and execution. This usually takes the form of a software development kit (SDK) , with an application programming interface (API)to drive the logic, plus, possibly, tools like a command line interface (CLI)to operate the logic from the edge device or any other terminal. We refer to the Edge (micro-)service operating logic as the edge runtime.

- **Container Facilities** build a container (such as Docker) with a trained MLM and possibly other Edge Framework artifacts, and store it in a Cloud registry for deployment on edge devices. Edge devices would then pull containers directly from the registry.

- **Communication Hub** create a messaging infrastructure among the different modules of the application. A popular communication strategy is based on events conveyed by lightweight protocols such as Message Queuing Telemetry Transport (MQTT): the reception of an event triggers an action corresponding to a workflow step or an operation.

- **Storage Facilities** store training data and several MLM versions in (possibly different) Cloud data warehouses. Data for the initial MLM are usually uploaded manually from a developer's workstation; subsequent data failing the inference are pushed to the Cloud from the edge device.

- **Machine learning Facilities** build and train a MLM in the Cloud. In the case of a supervised learning strategy, a human interaction is required to label training data. In case of forecasting, the label is known in the future.The Edge Framework signals to the Cloud, through the communication hub, the availability of new low-performance data, so that a learning round can be coordinated: the output of this process is a new version of the MLM.

And optionally, the following components:

- **Specialized Edge Hardware** edge hardware devices optimized for a given MLM implementation.

- **Control Room Facilities** any means to control and monitor all the above service components, such as dashboards and command line interface (CLI).

  - A dashboard is usually part of a graphical user interface (GUI): it should ideally allow operators to perform all edge-cloud operations, or provide a companion CLI covering any missing task. A dashboard can be Web-based or a separated application.

  - A CLI allows operators to perform tasks from a shell-like terminal. Ideally, once the platform is configured, the provided CLI would enable complete service orchestration (and possibly monitoring) via scripting.

3.2 AWS Greengrass

Amazon Web Services (AWS) [22] offers a complete solution to deploy services on edge devices. These services can be serverless with lambda technology, or containerized.
In Amazon Web Services (AWS), the components shown in Figure 3.1 are the following:

- **IoT Infrastructure Management** AWS IoT Device Management. A service, agnostic to device type and OS, for tracking, monitoring, and managing IoT devices. Deployments can be organized in device hierarchies; over-the-air (OTA) updates as well as auditing are supported; monitoring is supported via Amazon CloudWatch.

- **Edge Framwork** AWS IoT Greengrass. Extend AWS to edge devices so they can process locally generated data, even if disconnected. The edge application logic can be implemented either as serverless via AWS Lambda functions or as a (micro)service via Docker containers.

- **Container Facilities** Amazon Elastic Container Registry (ECR). AWS private Docker registry. No facility for building containers.

- **Communication Hub** AWS IoT Core (Message Broker). An MQTT-based messaging service, secured by encryption based on TLS version 1.2, that can work across the full AWS ecosystem, from Edge/IoT sensing services to Cloud storage facilities. It can also handle disconnected devices. Further processing is done via the EventBridge service.

- **Storage Facilities** Amazon S3. An object storage service well suited for both raw sensing data and MLM definition files. S3 can be considered a de-facto standard, because it is a widely adopted protocol outside the AWS' ecosystem.

- **Machine Learning Facilities** Amazon SageMaker. A data labeling (via the Ground Truth HITL-based service) and generic ML training service based on Jupyter notebooks, provided by either AWS Marketplace or app developers.

- **Specialized Edge HW** Snow Family. Also, qualified HW is available through the AWS Partner Device Catalog.

- **Control Room Facilities** AWS Management Console and AWS CLI.

3.3 Azure IoT Edge

Microsoft Azure IoT Edge [23] enables container deployment on Linux or windows devices.
In Azure, the components shown in Figure 3.1 are the following:

- **IoT Infrastructure Management** IoT Hub. Azure IoT Hub provides a cloud-hosted solution to manage IoT/Edge devices: provisioning, authentication, deployment overview, container monitoring, etc. It also offers a message routing service for edge-cloud communication based on either MQTT or AMQP.

- **Edge Framework** IoT Edge Runtime. This is composed of two runtime modules: Edge Hub and Edge Agent. The Edge Hub acts as a local proxy of the IoT Infrastructure Management (IoT Hub) and mimics its functionalities: it is for the IoT devices what the IoT Hub is for the Cloud. The Edge Agent is responsible for the management of user modules which compose the edge application code. All modules (runtime and user) are stored and deployed as Docker containers.

- **Container Facilities** Azure Container Registry. A private service based on the open-source Docker Registry 2.0. Any other public Docker registry can also be used. Partial Builder support with ACR Tasks.

- **Communication Hub** Provided by IoT Hub message routing services. Further Cloud processing of telemetry and application's messages can be done via either the Event Hubs or the Event Grid service.

- **Storage Facilities** Azure Storage. It covers different data type objects: blobs, files, queues, tables, and disks. The storage account provides a unique namespace to access data from anywhere over HTTP or HTTPS (RESTful).

- **Machine Learning Facilities** Custom Vision. An image classifier build and training service based on a limited set of model domains ("general", "food", "landmarks", or "retail"). A trained classifier (MLM) can be exported, as a Docker container, in one of several compact forms ("CoreML", "TensorFlow", "ONNX" or "Vision AI Dev Kit"), suited for use on mobile/edge devices with limited computing resources. Support for semi-automated labeling is also available.

- **Specialized Edge HW** None. Certified HW is available.

- **Control Room Facilities** Azure Portal and Azure CLI IoT. The IDE "Visual Studio Code" with an extension allows to deploy directly the containers to edge devices.

3.4 Google Cloud Platform

Google Cloud Platform (GCP), for now, only provide basic IoT [24] connectivity. For their Coral board [25] with an integrated Tensor Processing Unit (TPU) allowing accelerated ML inference, they recommend using their software partner Balena.
In GCP, the components shown in Figure 3.1 are the following:

- **IoT Infrastructure Management** Cloud IoT Core. A managed service to securely connect, manage, and ingest data from dispersed devices. It also provides a communication service based on the MQTT or HTTP protocols. No native orchestration service is provided.

- **Edge Framework** Unlike AWS, Microsoft Azure and SixSq Nuvla, Google does not provide a full-fledged Edge Framework component. In the GCP ecosystem, the Edge Framework is represented by the Cloud SDK and the native TensorFlow(Lite) library. The edge module can be containerized and deployed onto the edge devices either manually or via Balena.

- **Container Facilities** Trained TensorFlow (TF) models can be exported as Docker containers and stored in Google Cloud Storage. Apart from Edge Framework modules, no application code can be added to these containers; TensorFlow models can then be served to edge devices either manually via the standard Docker CLI. A pre-built CPU container that already has the whole environment to serve exported Edge models is stored in Google Container Registry.

- **Communication Hub** Pub/Sub. Edge devices can publish telemetry events, get configuration data, or set device state through either the MQTT or the HTTP protocol "bridge" of the Pub/Sub service. Messages can so be employed to trigger further Cloud processing, such as analytics and ML (re)training.

- **Storage Facilities** Cloud Storage. A RESTful object storage Web service with a proprietary interface. It has limited interoperability with S3.

- **Machine Learning Facilities** AutoML Vision. Three kinds of MLM are available for edge devices: single-label-based image classifier, multi-label-based image classifier and object detector with bounding boxes. All models are implemented in TensorFlow and, once trained, can be deployed on edge devices, such as the specialized Coral accelerators (see "Specialized Edge HW" below), smartphones, etc. A HITL-based labeling service is also available, although the AutoML API does not support it.

- **Specialized Edge HW** Coral accelerator. A series of Tensor Processing Unit-based (TPU) appliances running a GNU/linux-based OS. Coral features an ASIC optimized to accelerate TensorFlow-Lite-based inference models.
- **Control Room Facilities** Google Cloud Console and gcloud CLI.

3.5 Balena

Balena [26] focus on fleet management of edge devices. They provide a complete solution for build, deploy and monitor containerized applications. It is a closed solution delivered with a dedicated OS.
In Balena, the components shown in Figure 3.1 are the following:

- **IoT Infrastructure Management** Balena. A docker-based container application management platform, available as either a stand-alone open-source software stack (openBalena) for installation on premises or as a PaaS (balenaCloud), but with a different feature set. Devices are tightly integrated into a balena ecosystem via the balenaOS. BalenaCloud only provides basic OS/Docker remote log watching on its Web dashboard, and a minimal OS telemetry (no application) service.
- **Edge Framework** balenaOS. A Yocto linux-based OS optimized to run containers on edge devices. BalenaOS images are available for the most common edge HW architectures. The balenaEngine runtime is based on a stripped-down version of Docker's Moby.
- **Container Facilities** BalenaCloud provides a remote builder service and a container registry, whereas openBalena only provides local development builder tools. In the former cases, incremental "delta updates" are built; in the latter case, full docker images are built. In both cases, just a minimal image configuration (docker file) is needed; private Docker registries are also supported.
- **Communication Hub** Balena does not offer any messaging services for applications. It is up to the application developer to choose/implement one and ship it with an OS image update.
- **Storage Facilities** Balena does not provide Cloud storage for application data, though any S3-compatible solution can be used.

- **Machine Learning Facilities** No tools are available to build and train MLMs in the Cloud.

- **Specialized Edge HW** balenaFin, a professional carrier board for the Raspberry Pi Compute Module [17].

- **Control Room Facilities** BalenaCloud provides a full-featured proprietary fleet management Web dashboard. As this tool is not available for on-premises installation, openBalena users are left with the open-source CLI.

3.6 Nuvla

Nuvla [27] solution also focuses on fleet management and containerized application deployment. This is a solution that can be used on a device where the only prerequisite is to run docker and docker compose. The device can also run other services in parallel to nuvlabox.
In Nuvla, the components shown in Figure 3.1 are the following:

- **IoT Infrastructure Management** Nuvla. A container application management platform, available as either a stand-alone software stack for installation on premises or as a PaaS via Nuvla.io. All applications are packaged as Docker images, stored in a registry. Edge devices and Cloud Computing instances which are equipped with container management engines (like Docker and Kubernetes) can then be onboarded and used to provision container applications.

- **Edge Framework** NuvlaBox runtime software. A set of IoT- and Edge-specific microservices, used to transform any device into an NuvlaBox Edge device. This allows Nuvla to connect to and monitor each edge device individually. NuvlaBox microservices are containers, deployed through Docker under the control of the Nuvla cloud service. They provide facilities for: VPN-based networking with Nuvla and within the same edge device, MQTT-based internal messaging, application monitoring, security and discovery of attached HW components, such as network devices, GPU boards, etc.

- **Container Facilities** No specific service provided. Nuvla supports Docker with any public or private registry; orchestration relies on either Docker Swarms or Kubernetes via Nuvla(.io).

- **Communication Hub** The NuvlaBox provides an internal MQTT messaging system, for both the NuvlaBox components and any user applications running within the same internal network. For upstream communications with Nuvla, there is a rich HTTP-based RESTful API, as well as a Python client, which can also be used from within the NuvlaBox, via the NuvlaBox Engine Agent, or indepently, from any user application running on any infrastructure, provided the right user access credentials. Nuvla does not provide a private Communication Hub service, but lets you easily deploy one via its private app store.

- **Storage Facilities** Any S3-compatible solution can be used. On its part, Nuvla provides an S3 metadata catalog, with indexing and search functions.

- **Machine Learning Facilities** Unlike MS Azure, Amazon AWS and Google Cloud Platform, no tools are available to build and train MLMs in the Cloud. Any external service providing Docker containers for trained models and related application code can be integrated with Nuvla (see notes to Table 1).

- **Specialized Edge HW** None, but several edge hardware solutions are already certified to run the NuvlaBox software.

- **Control Room Facilities** Nuvla provides a full-fledged management Web GUI. No CLI is available, but a feature-rich API can be used to program any task via REST (language-agnostic) or Python calls.

3.7 Other Edge to Cloud solutions

Other solutions exist but have not been tested. They are briefly described here as a list according to the information available on their website.

- **BeeZeeLinx** [28] is a smart city software platform. It allows gathering IoT device data and displaying them in a dashboard. It provides a ways to manage asset, place the devices on a map and interact like switch on a street lamp. With the large number of data measured, CityLinx™ provides a unique powerful and natural query language to get the best out of these data. Finally, it provides a rule engine to detect abnormal situations and manage alarms.

- **EDGE IMPULSE** [29] is a platform to easily deploy machine learning model on edge device. It offers a beautiful interface for all steps, namely: device connected, data acquisition, Neural Network design, EON tuner (AutoML tools for optimizations), Retrain model, Live classification, model testing, and finally deployment.
  This solution is great to deploy MLM but is very specific, and you cannot deploy your own software.

- **Alibaba Cloud - Link IoT Edge** [30] allows you to deploy code on edge devices. the code can be in three different forms: Compute function (as a serverless function executed in the cloud, it can be executed on-demand and continuously), a container image, or local files. The platform also provides the infrastructure to route messages.

- **IBM Edge Application Manager (IEAM)** [31] is a platform where software developers can create and push container applications on container registry, before registering them on the management hub. Then edge node owner can deploy these applications on their devices. The documentation is not very explicit, and the service is aimed at advanced users of the IBM cloud.

3.8 Conclusion

The comparative study carried out in this chapter shows that the overall architecture of edge-to-cloud solutions is quite similar. Almost all are based on containerized application deployment. The work carried out in this master's project is part of the SWARM project [32]. Therefore, we opted to use the Nuvla/Nuvlabox solution. From a technical point of view, this open source solution seems to meet our needs.

4 | Devices deployment in Meyrin

4.1 Context

This chapter presents the deployment carried out at the "École des vergers" (school) in the municipality of Meyrin (canton of Geneva). This deployment is composed of:

- CLEMAP as grid edge device. The choice of this solution is driven by the LASAGNE project.
- Nuvla/NuvlaBox as edge-to-cloud solution. The choice of this solution is driven by the SWARM project.

The chapter is divided into three sections, with first a description of the micro-grid that will support our deployment, then the CLEMAP hardware (GED), and finally the Nuvla solution.

4.2 Micro-grid infrastructure

In Switzerland, two municipalities in the canton of Geneva (Meyrin and Chêne-Bougeries) are partners in the SWARM research project to deploy our devices in their buildings (private or public).

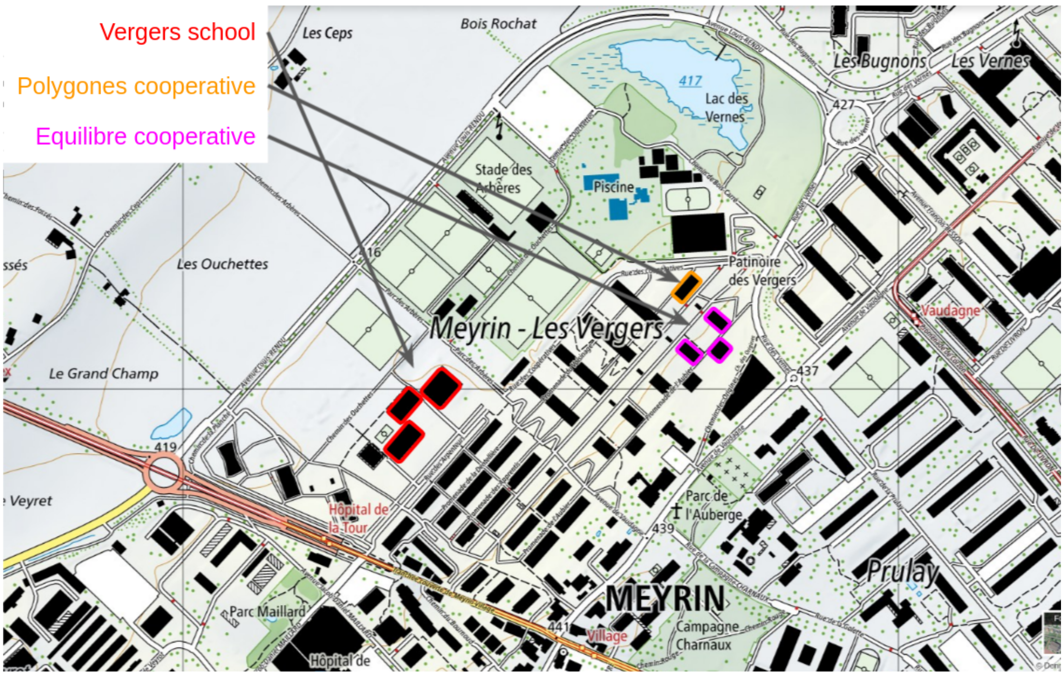

Figure 4.1 Verger eco-district in Meyrin

In Meyrin city, the newly built "Vergers" eco-district [33] offers a great opportunity to host our deployment. All buildings are self-generating and meet Minergie-A [34] standards. Before installing devices in cooperative building, a first deployment is made in a school (Figure 4.1), which belongs to the municipality. It is composed of 3 buildings:

- **School** (South-West building) with the classrooms.
- **Extracurricular** (Center building) with the canteen for the children's meal.
- **Gymnasium** (North-East building) for school children but also for sports competitions.

Distributed in the three buildings, seven devices are installed to measure various consumers such as light, kitchen, refreshment, ventilation, heating.
Among these devices, four are in the basement and are connected to the Internet through a cable. For the other three, they are installed on the ground floor and are covered by the 4G network. To speed up the installation and reduce the cost, we used 4G modems for these three.

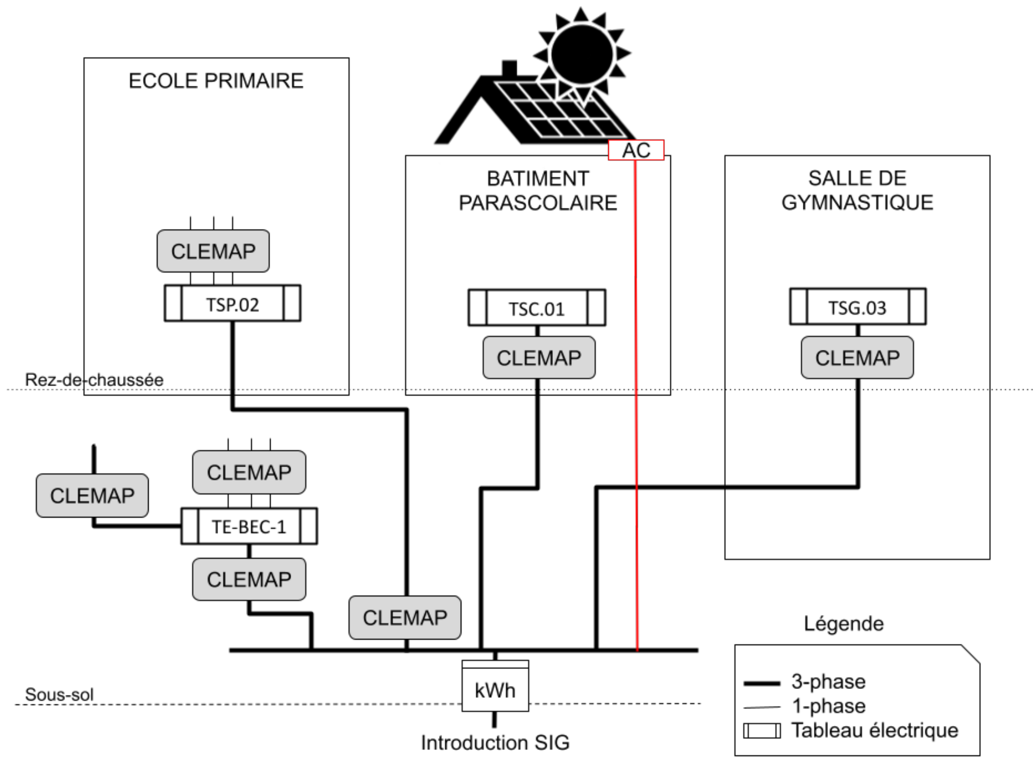

Figure 4.2 Electrical diagram of the GED installation

The Figure 4.2 shows the installation of the CLEMAP GED on the school's electrical diagram. When the CLEMAP is shown under the electrical panel, it measures the consumption of all the equipment connected to the panel. When the CLEMAP is on three single-phase lines, each of its current sensors (Figure 4.4) is connected to a single-phase loader. Other electrical panels exist in the school but are not shown here.

Finally, the list below describes, the consumption measured by the CLEMAP, according to the reference of the electrical panels.

- **TSP.02** One CLEMAP for three different lightings and one CLEMAP for the global consumption including lighting, wall sockets in classroom ...
- **TSC.01** This CLEMAP measures kitchen appliance for the canteen. This includes deep fryer, oven, cooking table, fridge, dishwasher and lighting ...
- **TSG.03** This CLEMAP measure wall socket and refreshment stand including refrigerated cabinet, meeting room, lighting, dishwasher ...
- **TE-BEC-1** This electrical panel is located in the boiler room. A CLEMAP measures the tank water heater in three phases, and another CLEMAP measures three single-phase ventilation. The last CLEMAP measures the global consumption.

4.3 CLEMAP smart meter

As explained in previous chapter, CLEMAP offers a device which measure power consumption/production, monitor and classify appliance usage. This device has been chosen for this deployment because the company involved in the LASAGNE [35] project. This device will be improved with forecast module for predicting future usage of power.

4.3.1 Hardware

The device is designed to be installed in an electrical cabinet on a DIN rail. On one side the three three-phase voltages (L1, L2 and L3) and the neutral are connected. This allows the device to be powered, and the voltage of each phase to be measured. On the other side three current sensors are connected. The sensor (Figure 4.4) forms a ring that is placed around the electric cable to measure the radiated magnetic field. Finally, an Ethernet cable is connected to allow the device sending data to the cloud. The internet connection can also be done by Wi-Fi, but the wire connection is preferred for better availability, and less sensitive to disturbances. Voltage (V) and current (A) measurements are used to calculate active (W), reactive (VAr) and apparent (VA) power. The φ angle and power factor are also calculated. The power in alternating mode is

explained in the Appendix A
The device consists of an electronic board developed by CLEMAP, allowing connecting the sensors, and a Raspberry Pi [17] connected through its GPIO ports, to process data and send it to the cloud.

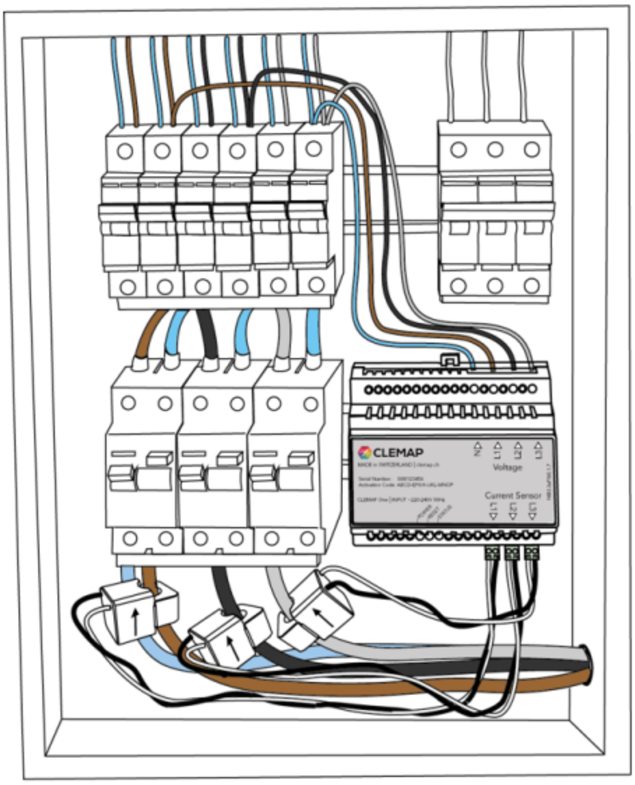

Figure 4.3 Installation diagram in the electrical cabinet

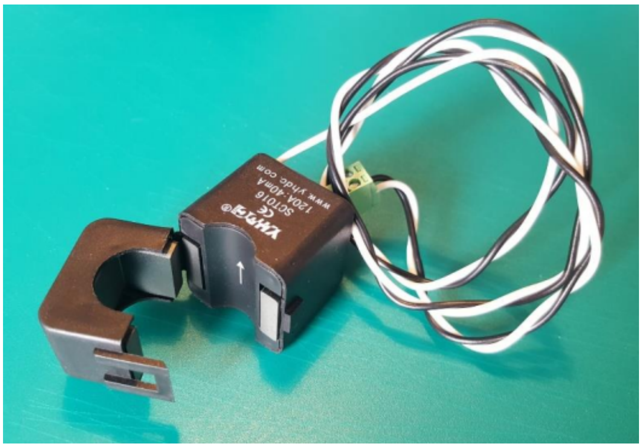

Figure 4.4 Current sensor

4.3.2 Software

The CLEMAP software collects measures at a frequency of 12Hz. These measures are used to classify households' appliances. A report is also sent to the cloud. The initial functioning is shown in Figure 4.5. The aim of the project is to process the measures within the GED. To minimize the modifications in the CLEMAP software, the MQTT protocol is used to send the measurements to our deployed application. Measured data are sending at two frequencies: every ten seconds and every minute. The content of each message is encoded in binary format, and sent on two different MQTT topics. The contents of the messages are detailed in the Table 4.1a and Table 4.1b. In these tables, the value <n> can be 1, 2 or 3 and represents the phase of the three-phase current. To understand power value measured in AC, please refer to Appendix A.

Label	Type
feature_type	int
timestamp	int
sensor_id	ObjectId
phase	int
v_l<n>	float
i_l<n>	float
s_l<n>	float
p_l<n>	float
q_l<n>	float
pf_l<n>	float
phi_l<n>	float
avg_energy_l<n>	float

(a) one minute data

Label	Type
feature_type	int
timestamp	int
sensor_id	ObjectId
phase	int
p_l<n>	float
q_l<n>	float

(b) ten seconds data

Table 4.1 Composition of the messages sent by MQTT

In order to train our forecast algorithm, the data must first be recorded. For this, a containerize program subscribes to the two MQTT topic. The payload is saved in a file, and upload to a storage service. For the moment, the service is located in multiple regions in European Union, for simplicity in the context of test and development. For privacy reasons, it is planned to use a Swiss service location for Swiss deployment.

To have an overview, Figure 4.5 represents the architecture proposed by CLEMAP of the product they sell. With the software modifications made on the device, we obtain the following Figure 4.6 and Figure 4.7. The first one shows the data acquisition phase, and the second one for the inference phase to predict the future consumption. The forecasting result is not used for the moment but it will be used in the future as input of high-level end-user application to negotiate with the neighbors and/or to adapt the consumption with the aim of a flexible use.

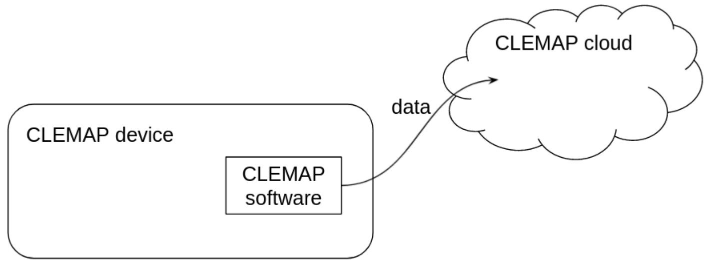

Figure 4.5 Clemap architecture

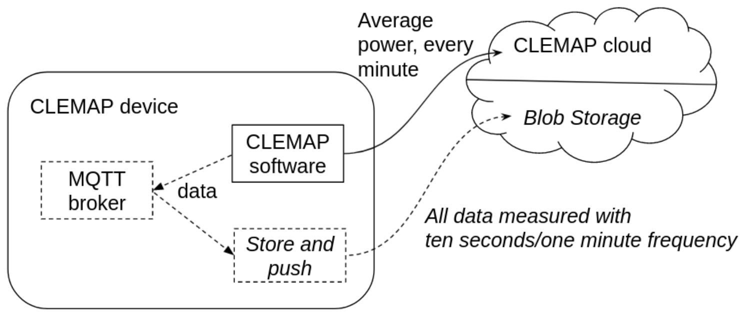

Figure 4.6 Clemap architecture, collect data for learning

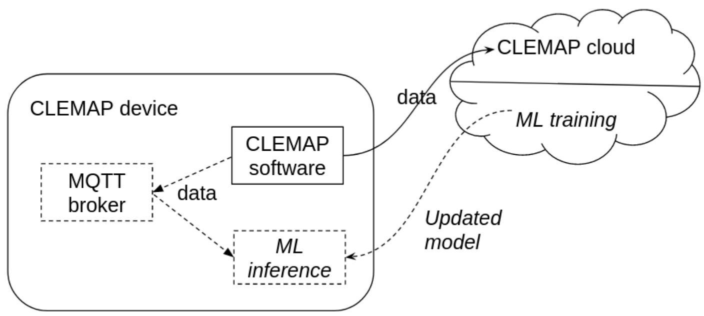

Figure 4.7 Clemap architecture, use realtime data for inference

4.4 Nuvla Box

As described in the previous chapter, Nuvla offers an environment for edge computing to monitor and deploy code on devices. Among all the possible ways to manage edge devices, Nuvla has been chosen because SixSq (the company that develops Nuvla/NuvlaBox) is the main partner of the SWARM [32] research project.

To turn a device into an edge device connected to Nuvla, you need to install the NuvlaBox Engine (NBE) in this one. This is done by installing docker and docker-compose to launch NuvlaBox engine, schematized in the Figure 4.8. The different elements of the NuvlaBox Engine are detailed in the Table 4.2.

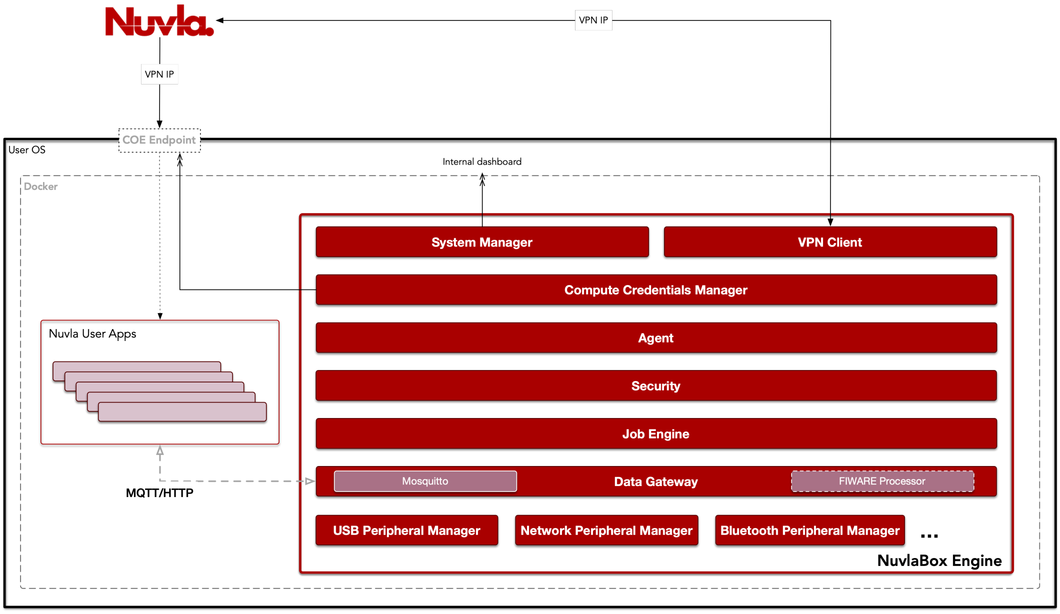

Figure 4.8 NuvlaBox engine architecture [36]

The NuvlaBox Engine is composed of many components, and the Raspberry Pi 3 [17] located inside the CLEMAP device has limited resources That's why the peripherals manager modules (optional) are not used, and the security component has been removed because it's not necessary to work.

The Data Gateway container is accessible only from the virtual network created by the Nuvlabox "*nuvlabox-shared-network*". But the software developed by CLEMAP does not have access to this network. So, a Container is then used to link the host network with the network created by the NuvlaBox through port 1883. This proxy container allows communicating with the MQTT protocol.

NBE Component	Description
AGENT	The agent is responsible for the activation, commissioning and monitoring precedures
SYSTEM MANAGER	The system manager acts as a watchdog. It ensures that all containers are in a healthy state. If it finds an error, it makes sure to repair it automatically.
COMPUTE CREDENTIALS MANAGER	Also named compute-api in the device, this component makes sure all synchronous communications are secured.
VPN CLIENT	Establishing a secure VPN tunnel with SixSq's VPN server thanks to an OpenVPN configuration set by the agent. This ensuring that it can be accessed from anywhere without the need to forward a port or have a public ip.
SECURITY	the security component regularly checks the edge device for vulnerabilities referenced in public databases (like CVE)
DATA GATEWAY	The data gateway allows users to simplify the sending of messages and the acquisition of sensor data. Mosquitto is an mqtt broker, and optional FIWARE processor, allows the validation of the data schema.
JOB ENGINE LITE	It is a sporadic component that allows to execute tasks asynchronously, even with a lost connection.
PERIPHERAL MANAGER	These are optional micro-services which perform automatic detection and categorization of peripherals that are or can be connected to the edge device. There is peripheral manager for usb, bluetooth, modbus, gpu, network and others

Table 4.2 NuvlaBox Engine description

4.5 Conclusion

The deployment was challenging because it involved several stakeholders. The deployment is functional but suffers from some weaknesses. The low speed provided by the 4G connection (limited by the operator's subscription) does not allow the deployment of containers by NuvlaBox technology. The performance of the Raspberry pi 3 brings great risk of infrastructure malfunction.

The two next chapters will cover machine learning forecasting of power usage. The inference module is intended to be used on deployed GED with local data.

5 | Forecast algorithm for energy consumption

5.1 Context

This chapter deals with machine learning algorithms allowing making a prediction on a time series. The first section describes the design of machine learning algorithms, with methods to evaluate their performance, the problems of an under-trained or over-trained algorithm, and finally the particularity of a dataset in the form of a time series, i.e. values measured over time with a fixed interval. The following sections detail the functioning of the algorithm selected for our problem: time series forecasting. These algorithms are: XGBoost, LSTM, DeepAR, Temporal Fusion Transformer, and Prophet. They have been chosen for their different functioning.

5.1.1 Artificial Intelligence

Artificial intelligence (AI), which is based on mathematical and statistical approaches to give computers the ability to "learn" from data, i.e. to improve their performance in solving tasks without being explicitly programmed for this. The learning algorithms are characterized by the learning mode used.

- **Supervised learning** to use this algorithm, you must first have a labeled data set. Then, in a first time the algorithm learns. It is a recurrent process where at each step the result of the algorithm is compared to the label thanks to a cost function. In order to evaluate the algorithm, in a second step, we apply it to new data, not used during the training, and we compare the result with the given label.

- **Unsupervised learning** Unlike the previous point, here the training data does not have a label. The algorithm then manages to group the data based on the distance between them.

- **Semi-supervised learning** This learning mode is between the two previous ones. The training dataset contains both labeled and unlabeled data.

- **Partially supervised learning** Unlike semi-supervised where some labels are missing, here can have an indefinite number of labels.

- **Reinforcement learning** This learning method lets the algorithm explore solutions and get a reward if its choice was right.

- **Transfer learning** Allows to use an existing model and to train it with your own dataset. The data used for the first training of the model must be similar to the current data.

Machine learning algorithms can be defined in four categories according to their purpose.
Classification consists of placing a label on a data. This label is known during training and is the output value of the algorithm when used. We can take the example of an algorithm that predicts the sex of a child at birth, whose label would be boy or girl.
A **Regression** algorithm will not have a label as output value but a data value within a defined range. We can take the example of an algorithm that predicts the price of an apartment based on several criteria such as the area.
A **Clustering** algorithm is used to group data that are similar. For this purpose, the distance between each data is measured. It is an unsupervised algorithm that allows obtaining a classification, by placing a label on each group create. This technique allows obtaining a segmentation that a human would not have easily found. For example, the customers of a store can be stigmatized in relation to their purchasing habits.
Learning by **Association rule** allows determining data that go well together. For example in an e-commerce site, an algorithm of this type will propose products according to the content of the shopping cart.

5.1.2 Machine learning algorithm evaluation

A supervised algorithm is **evaluated** by comparing the result obtained with the defined label. For a classification we obtain a confusion matrix (Table 5.1). If the problem has more than two labels, the number of rows and columns must be increased accordingly. In the diagonal, we obtain the number of times the algorithm did not make a mistake with TP and TN acronym. In the other boxes, we get the number of times the algorithm was wrong, with FP and FN acronym.

		Predicted		
		Positive	Negative	Total
Actual	Positive	a (TP)	b (FN)	$a + b$
	Negative	c (FP)	d (TN)	$c + d$
	Total	$a + c$	$b + d$	N

Table 5.1 Confusion Matrix

From this matrix, several values can be calculated to evaluate the model. The equations are shown below.
Accuracy is for : Overall, how often is our model correct?
Precision is for : When the model predicts positive, how often is it correct?
Recall helps when the cost of false negatives is high.
F1 is an overall measure of a model's accuracy that combines precision and recall. It produces a value between 0 (total failure) and 1 (considered perfect).

$$Accuracy = \frac{TruePositives + TrueNegatives}{TotalExemples}$$

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

$$Recall = \frac{TruePositive}{TruePositives + FalseNegatives}$$

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

For regression algorithms, it is not possible to use a confusion matrix because the output is not discrete but continuous. It is necessary to calculate the distance with methods like MSE or RMSE to accentuate the error on the big distance, or MAE to have a linear error compared to the distance. It is also possible to use the variance to identify the correlation between the predicted value and the real value, for this we use the coefficient of determination R^2. Formulas are expressed below. The scikit learn [37] python library, contains metrics and scoring to quantify the quality of predictions.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad ; \quad RMSE = \sqrt{MSE}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

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

Where n is the number of observations, y_i is the actual value, $\hat{y}_i$ is the corresponding predicted value and $\overline{y}$ is the average of the measurements.

5.1.3 Under fitting and over fitting

In these algorithms that learn on data, we try to avoid being confronted with two opposite problems: **under fitting** and **over fitting** (example in Figure 5.1). This happens either if the algorithm has not learned enough or if it has learned too much. In order to avoid over fitting, the algorithm is never tested on the same data as the one used for training. In order to evaluate an algorithm avoiding over fitting problems it is possible to use the k-fold cross validation method. The labeled dataset is cut in k pieces. Then we train with $k-1$ pieces and we test on the last piece. We repeat by changing the piece used for the test.

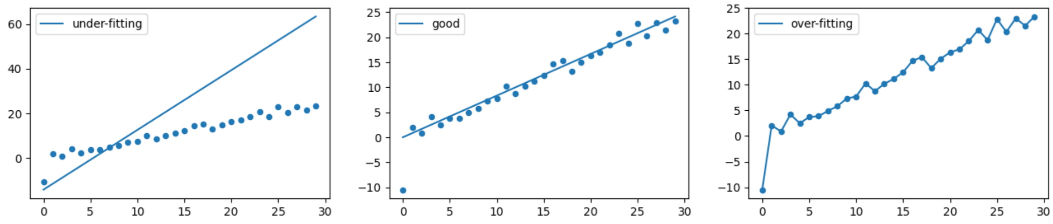

Figure 5.1 Under fitting, expected, and over fitting

5.1.4 Time series Forecasting

We refer to **univariate** machine learning model when we use only one variable that varies in time, for example only the apparent power consumed. On the contrary, when we use several variables, for example apparent and reactive power, it is a **multivariate** model. We can use the term horizon to define the output of a time series forecasting algorithm. This horizon can be near (next quarter of an hour) or far (next week, month). In some cases, depending on the algorithm used, it is possible to make a multi-horizon prediction.

The form of the evolution of the value over time can be defined according to the following criteria, shown in the Figure 5.2. The **trend** shows through an almost linear function how the value evolves over time during a long period of time. It can increase, decrease, or be stable. The **seasonality** of a time series is a regular pattern that repeats each x time periods. For example, if we consider a clear sky, the value of the solar production is repeated over the days between sunrise and sunset. The **stationary** criterion is determined by measuring the mean, variance and covariance that do not vary over time. The mean evolution is considered null when the trend is null. To identify if a time series is stationary, we can either perform a visual test by representing the data as a graph, or use one of the two following statistical tests: Augmented Dickey-Fuller (ADF) or Kwiatkowski-Phillips-Schmidt-Shin (KPSS).

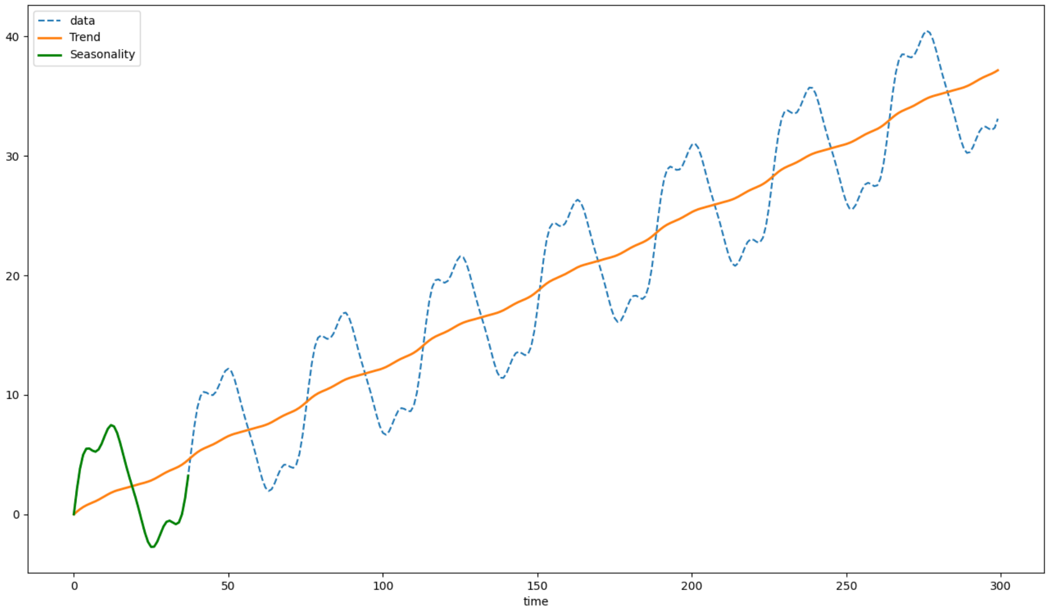

Figure 5.2 Trend and Seasonality

The following section is a selection of relevant algorithms for value prediction with a time series. These algorithms belong to the supervised learning category because the labels are known, and it is a regression because the goal is to predict the future consumption / production which is a linear value.

5.2 XGBoost

eXtreme Gradient Boosting (XGBoost) [38] is an ensemble method with the particularity of being easy to use, fast, and good choice for an algorithm to start with. It also has good performance. It can be used for classification, regression or ranking problems. It consists of decision trees set, so is faster to train than a Neural Networks. It is a good candidate for use on devices with limited resources because for a prediction, it needs little resources both in computation and memory.

During the learning step, the algorithm aims to optimize an objective function. This function is composed of training loss (L) and regularization term (Ω).

$$obj(\theta) = L(\theta) + \Omega(\theta)$$

The training loss measures how predictive our model is with respect to the training data. The regularization term controls the complexity of the model, which helps us to avoid over fitting.

At each learning step a new tree is built to improve the previous one. This tree is added to the model, resulting in a decision tree set. Each node of the tree is a comparison with a value of input vector. Each leaf is a probability. Each probability of each tree is grouped to give the output value.

The Figure 5.3 illustrates the model. The input dataset is introduced in each tree. The result of each tree is grouped in a function to give an output value.

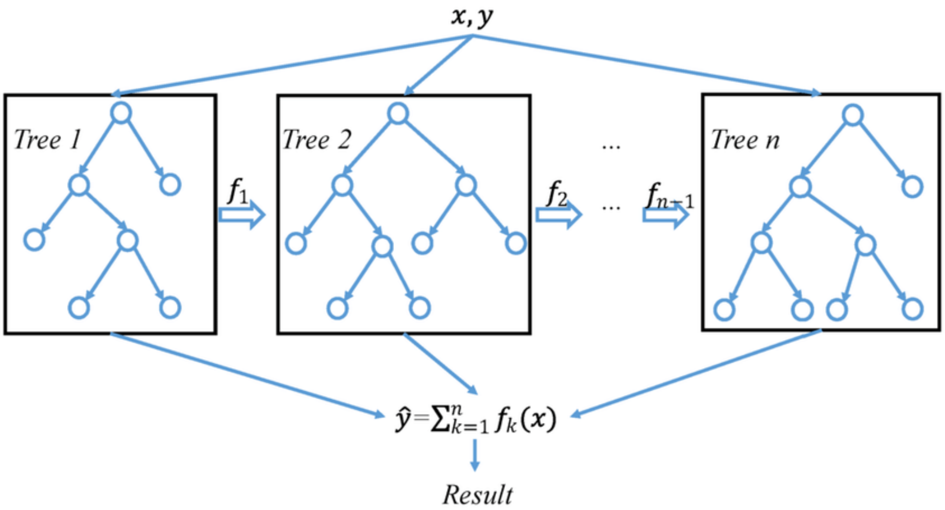

Figure 5.3 XGBoost [1]

5.3 LSTM

Long Short-Term Memory (LSTM) is an algorithm that uses Neural Networks (NN), and more specifically Recurrent Neural Networks (RNN). Unlike the feed-forward neural network (ANN family) where the data progresses unidirectionally from input to output, in the recurrent neural network, the data are looped. This implies a risk of vanishing or exploding gradients. The term of the first goes to zero exponentially fast, while second goes to infinite.

LSTM take as input following values: Samples, Timesteps and Features. Samples are a vector of continuous value. Timesteps are the numbers of time steps in the samples vector. Features are the number of features in every time step. to summarize, the input is a fixed size sequence of one or more time series.

The Neural Networks process the input sequence by iterating through it and maintaining a state containing information relative to what it has seen so far. The memory state inside LSTM allows the current and past value to be taken into account. The further back in time the value is, the less important it is. This memory state is reset between processing two different independent sequences.

[1] Source: https://www.researchgate.net/figure/A-general-architecture-of-XGBoost_fig3_335483097

The Figure 5.4 illustrates the model. In Blue, we have the Samples input dataset. The green cell is repeated according to the number of time steps. Inside each cell, Neural Networks layer is in yellow and point wise operation in red. The output values are on top of the figure.

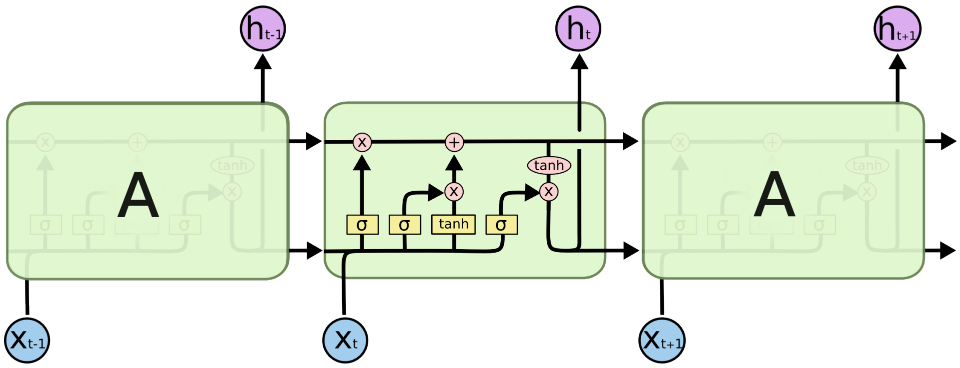

Figure 5.4 LSTM [2]

5.4 DeepAR

Deep Auto-Regressive (DeepAR) [39] is an algorithm developed by AWS for time-series forecasting using Recurrent Neural Networks (RNN). It is included in the sagemaker library which can only be run on an AWS instance. Amazon SageMaker is a fully managed service that provides the ability to build, train, and deploy models quickly. It is a web-based visual interface that makes ML development steps easier and faster at a lower cost. Amazon SageMaker notebooks are Jupyter notebooks [40] in an EC2 instance dedicated to run your environments, and codes for feature engineering to building a model.

An implementation of this algorithm is also available in the pytorch-forecasting library, based on the article *DeepAR: Probabilistic forecasting with autoregressive recurrent networks* [41]

As for LSTM, DeepAR uses Recurrent Neural Networks. During training, DeepAR accepts a training dataset of one or many time series and an optional test dataset. The algorithm automatically creates features to obtain seasonality (day of the month and day of the year). During learning, DeepAR take randomly several training example in the dataset (Figure 5.5). Each training example consists of a pair of adjacent context and prediction windows with fixed predefined lengths.

The output of the algorithm is a probability i.e. chance that the output value is lower/superior to an output value.

[2] Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[3] Source: https://docs.aws.amazon.com/sagemaker/latest/dg/deepar_how-it-works.html

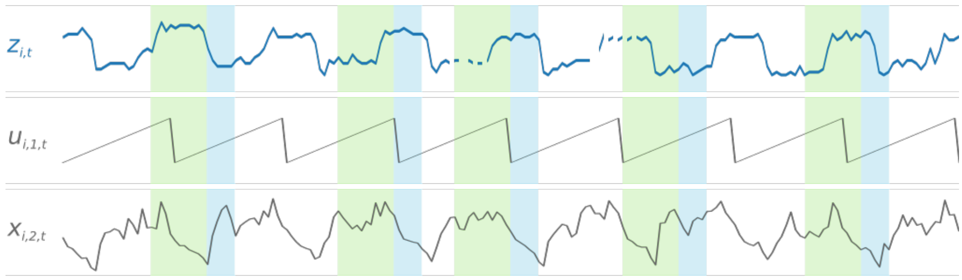

Figure 5.5 DeepAR [3]

5.5 Temporal Fusion Transformer

Temporal Fusion Transformers (TFT) [42] is a Deep Neural Network developed by Google for interpretable multi-horizon time series forecasting. It is an attention-based model for high-performance multi-horizon forecasting.

This model uses recurrent layers for local processing and self-attention layers for long-term dependencies. In addition to time series data, it accepts static time-invariant value know in the past, and also in the future.

As explain in the paper [42], the major constituents of TFT are :

- **Gating mechanisms** to skip unused components of the architecture, providing adaptive depth and network complexity to accommodate a wide range of datasets and scenarios.
- **Variable selection networks** to select relevant input variables at each time step.
- **Static covariate encoders** to integrate static features into the network, through encoding of context vectors to condition temporal dynamics.
- **Temporal processing** to learn both long- and short-term temporal relationships from both observed and known time-varying inputs. A sequence-to-sequence layer is employed for local processing, whereas long-term dependencies are captured using a novel interpretable multi-head attention block.
- **Prediction intervals** via quantile forecasts to determine the range of likely target values at each prediction horizon.

The Figure 5.6 is an illustration of multi-horizon forecasting. The model uses static covariates, past-observed and apriori-known future time-dependent inputs. This heterogeneity of data sources together with little information about their interactions makes multi-horizon time series forecasting particularly challenging [42].

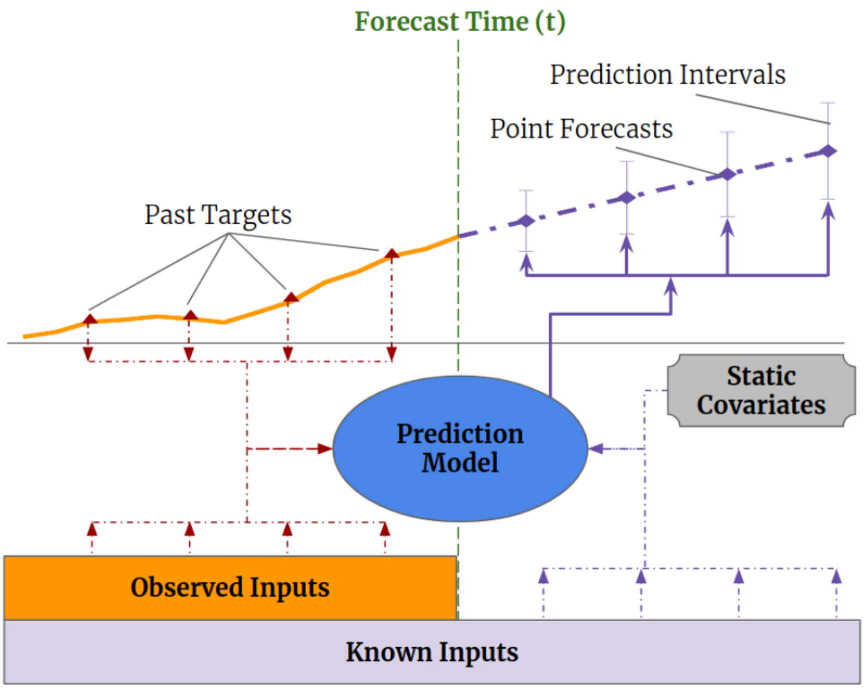

Figure 5.6 Temporal Fusion Transformers [4]

5.6 Prophet

Prophet [43] is an algorithm developed by Facebook [44]. It proposes an easy to use algorithm to get an overview of the problem. The algorithm is essentially based on the seasonality of the data to produce a multi-horizon forecast. It takes as input the timestamp and the measured value. After analysis of the data, it provides a prediction on the requested time window.

The algorithm uses change point to detect a change in trend. At the beginning, it uses a large number of possible places where the rate can change in the first 80% of the dataset. After learning, only the most relevant are kept, and can be visualized in a plot.

[4]Source : https://arxiv.org/pdf/1912.09363.pdf

The seasonality is extract thanks to Fourier transform. If the order does not match the data, it can be adjusted. By default, the algorithm fit daily, weekly and yearly seasonality. But custom seasonality can also be defined. Finally, holiday and special events can be defined, for example Black Friday for a forecast of the number of sales of a store.

The Figure 5.7 represents a prediction of the whole year 2016, with data (black dots) from 2008 to 2016 as input. In dark blue is the trend of the data, and in light blue is a probability that the data is in this space.
The Figure 5.8 is the trend, weekly and yearly seasonality of the same dataset. The probability space in the trend curve increase accordingly to the distance of the prediction.

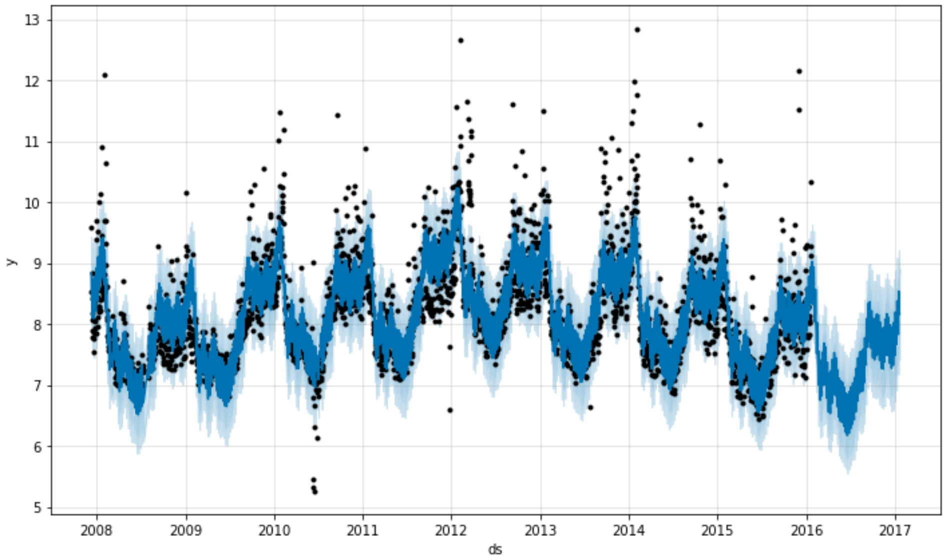

Figure 5.7 Prophet forecast [5]

[5]Source : https://facebook.github.io/prophet/docs/quick_start.html
[6]Source : https://facebook.github.io/prophet/docs/quick_start.html

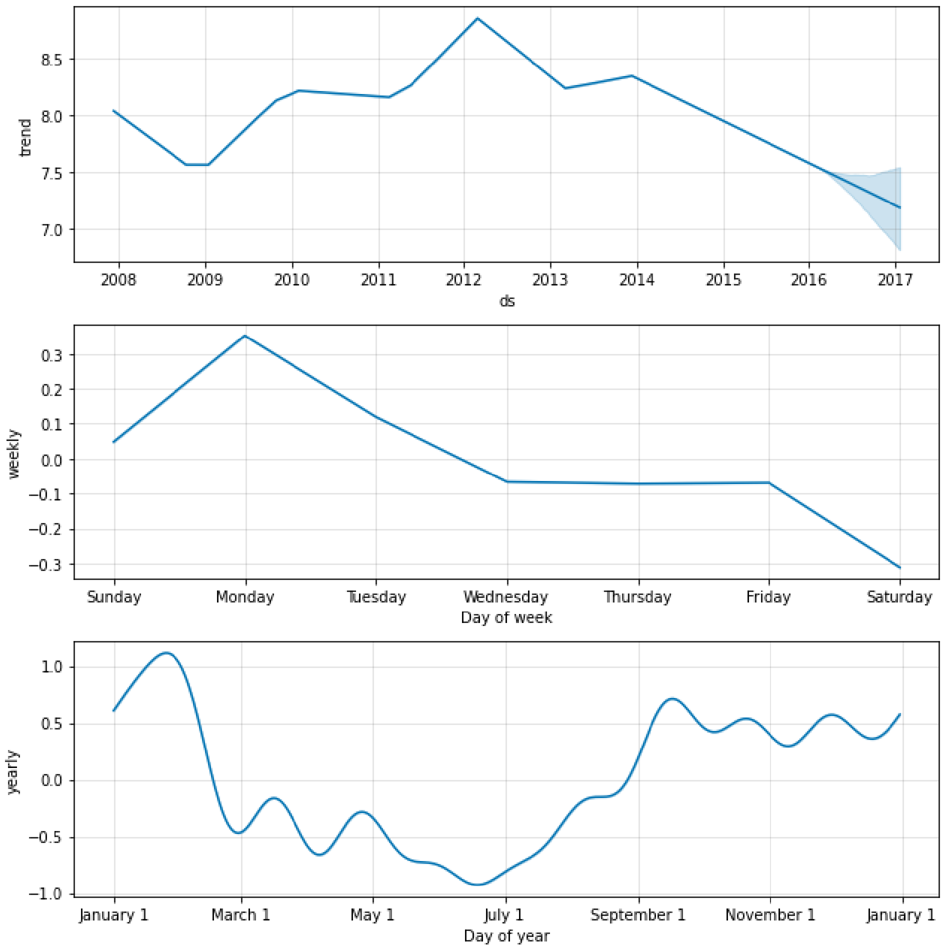

Figure 5.8 Prophet seasonality [6]

5.7 Conclusion

These algorithms have been tested and the results can be found in the next chapter. They are not all the same complexity, and in addition to the performance results obtained, the computational power needed to perform an inference must be considered before using them on GED.

6 | Forecast algorithm with Groupe-e dataset

6.1 Context

Before deploying forecast module on grid edge device, it is necessary to test them in order to deploy the most efficient one in terms of prediction quality and necessary computing power.

This chapter is divided in two sections. The first describe the dataset used. The second shows the results obtained on this dataset with the different algorithms seen before.

6.2 Dataset

For the SWARM project, Groupe e [45] shares the consumption/production data of their client. The details of this dataset are described below. Groupe e is a Swiss electricity producer and distributor. It serves the inhabitants of the cantons of Fribourg, Neuchatel and Vaud.

6.2.1 Grid data description

Distribution grids are organized in what could be best described as topological trees, interconnected in open or closed loops. The trees and loops then span different voltage domains, from lower to higher typically low voltage LV, medium voltage MV and high voltage HV [46].

The HV grid is commonly a closed loop, ensuring the electricity supply between regions and large areas. The MV grid is commonly made of distinct trees, originating from a high-to-medium voltage transformation substation. Interconnections may exist between distinct MV trees for grid recovery purposes. The LV grid is commonly made of distinct trees, originating from a medium-to-low voltage transformation substation, which represent nodes of the MV grid. Interconnections may exist between distinct LV trees for grid recovery purposes. At the LV level, distribution pillar boxes make the intermediary nodes of the LV trees, which then terminate in service points. For each LV service point, there will be one or several clients associated, with either consumption or production information, or in some cases both

The data provided by Groupe-e is in the form of a file in csv format. Among these files we have metadata information and power usage consumption (see subsection 6.2.2).

[1] courtesy of Groupe-e

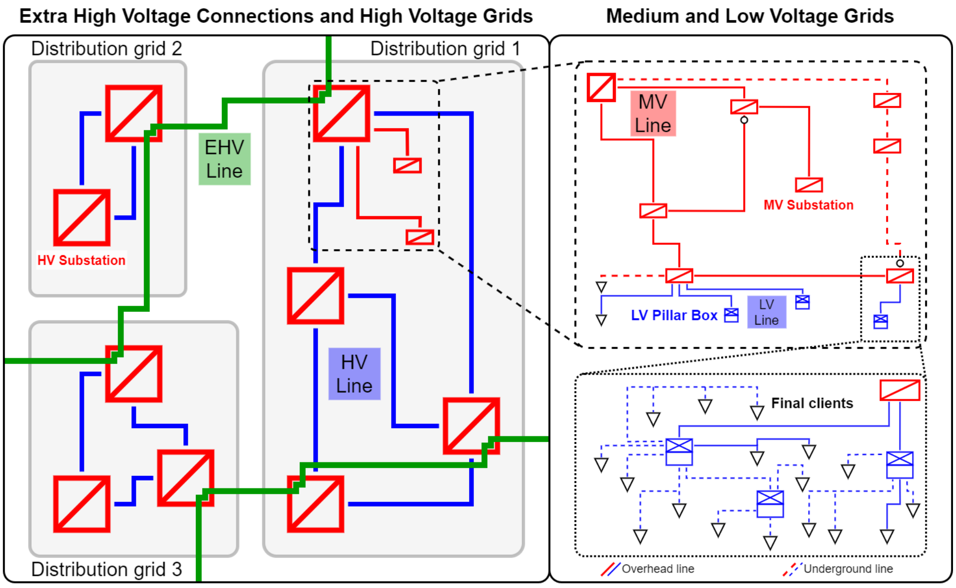

Figure 6.1 Grid topology [1]

The following list shows different files containing the grid metadata. The information is indexed between them by the *service point* which corresponds to the connection point with the main power grid. The last file (not in the following list) connects the device number that reads the power consumption, with the service point to which it is connected.

- **Topology** : In addition to the reference of the electric transformers allowing to pass from the high tension, towards medium tension, towards low tension, we find fuse rating of the service point, the location area and the building type supplied by the service point.

- **Client information** : Building type for the client and an operator filled description of the client.

- **Production** : If the client has solar, hydro or natural gas generation, the date of installation and the power of the equipment are indicated.

- **Disruptive** : Some devices (such as heat pumps, elevators or charging stations) disturb the stability of the network when in use. These are listed with their active/apparent power rating and current rating.

6.2.2 Metering data description

The two files shown in the following tables contain the values measured by the measuring devices. In Table 6.1, we can see that for each measuring device, we have the voltage and current values for the 3 phases (There is no guarantee

that the phase ordering is observed from one device to the other). We also have active and reactive power flowing in and out. The measured incoming power corresponds to the power consumed by the user, and the outgoing power corresponds to the power produced by solar panels (or other source of electricity) which is injected into the network, because the production is greater than the consumption.
In order to simplify these measurements, a second file represented by Table 6.2 contains only the difference between the incoming and outgoing active power. It is this value that is billed to the customer. If this value is positive, the customer buys electricity from the supplier (in this case Groupe-e) but if it is negative, the customer sells his surplus to the supplier.

The duration between each sample is fifteen minutes, it is a duration imposed by Art. 8a of "Ordonnance sur l'approvisionnement en électricité" (OApEl) [47]

Column	Information
DATETIME	The measure timestamp, in ISO-8601 format
<device ID>_IL1	Measured current for the first phase of the metering device in ampere (A)
<device ID>_IL2	Measured current for the second phase of the metering device in ampere (A)
<device ID>_IL3	Measured current for the third phase of the metering device in ampere (A)
<device ID>_UL1	Measured voltage for the first phase of the metering device in volt (V)
<device ID>_UL2	Measured voltage for the second phase of the metering device in volt (V)
<device ID>_UL3	Measured voltage for the third phase of the metering device in volt (V)
<device ID>_Pin	Measured flowing in active power in kilo watt (kW)
<device ID>_Pout	Measured flowing out active power in kilo watt (kW)
<device ID>_Qin	Measured flowing in reactive power in kilo volt-ampere reactive (kvar)
<device ID>_Qout	Measured flowing out reactive power in kilo volt-ampere reactive (kvar)

Table 6.1 Pre-processed data

Column	Information
DATETIME	The measure timestamp, in ISO-8601 format
<device ID>_p	Active power balance measured in kilo watt (kW)

Table 6.2 Pre-processed data, only active power balance

6.2.3 Data overview

The figure 6.2 gives an overview of the power consumption of the first five device of the provided dataset over the total period (January to September 2021). In total, there are 1179 devices. We can see that the measurements of some days are missing. The Table 6.3 shows the number of devices for each month that have a gap higher than 1h, 4h, 6h, of missing data. We notice that for the second device, the consumption is higher in winter than in summer. We can assume that this house uses electric heating. The fourth device has values close to zero for two and half months, the house was probably unoccupied. From this figure, we can see two types of appliances. Those that consume little but over a long period of time (appliances on standby, light, fridge ...), close to zero on the graph. Those that consume a lot but over a short period of time (electric stove, oven, kettle...), peak on the graph. The last device on the graph have fewer peaks than the other.

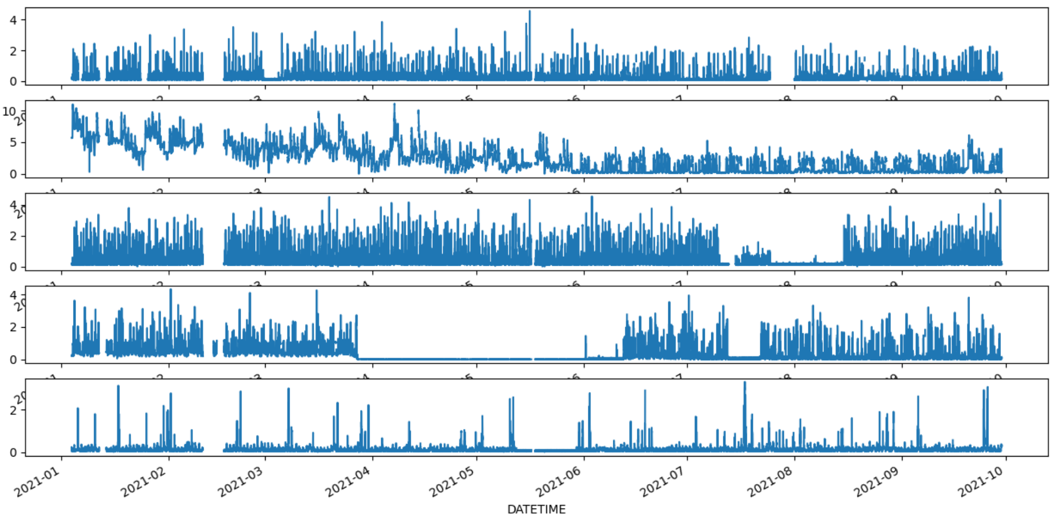

Figure 6.2 Five first devices

Number of dev with gap higher than:	1h	4h	6h
January	738	737	737
February	1170	1170	1170
March	96	93	93
April	105	95	95
May	1169	1169	1169
June	286	274	273
July	293	287	286
August	96	88	88
September	105	91	90

Table 6.3 Number of devices with data missing, month by month

6.3 Experimental result

The dataset we use contains measurements every fifteen minutes. Two types of predictions are then possible: the next unique measurement, or a series of measurements over a defined period (for example a day or a week). Obviously, the further into the future the prediction is, the lower the confidence index is. Depending on the type of prediction desired, we will not use the same algorithm.

To compare the different algorithm, the same data was applied to each algorithm. Only the power is used, with for each prediction the last 96 values (corresponds to 24 hours) as input, and the power consumed in the next 15 minutes as output. XGBoost does not accept missing data during training, so we can't use the whole dataset. The tests are based on the months of March and April where only 95 devices have missing data with an interpolation to fill the gaps up to 4 hours.

6.3.1 Comparison on one device

The first experiment consists in comparing the different algorithms (XGBoost, LSTM, TFT and DeepAR) on a single device using the same dataset. A baseline model was developed in order to have an idea on how well the other models will perform. The metrics used to compare performance are RMSE and MAE on the raw data in kilo Watt.

In Table 6.4, the RMSE and MAE values obtained from different models are presented. One can easily notice that all models are performing better than the baseline.

Model	Device	RMSE (kW)	MAE (kW)
Baseline (Moving Average)	425766_p	2.2562	1.930
LSTM	425766_p	0.804	0.49
XGBoost	425766_p	0.8043	0.6012
DeepAR	425766_p	0.9330	0.7898
TFT	425766_p	1.0767	0.793
Prophet	425766_p	1.1814	0.9047

Table 6.4 Results of predictions, based on different models

The following figures shows the results in graph form, presented in the Table 6.4, for the "425766_p" device. In the Figure 6.3, the top graph shows the training dataset, and the bottom graph the test dataset. The other figures, except for Prophet present only the test dataset. Except for LSTM, algorithms have difficulty adapting when the new data are not in the training range. In fact, for this device, since the middle of April 26, the value is below 2, while it is always higher in the training dataset.

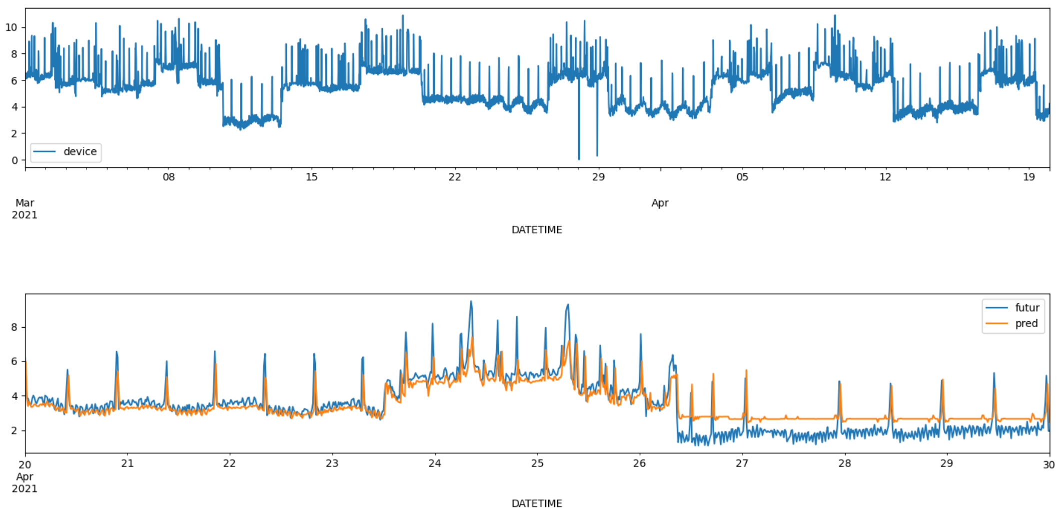

Figure 6.3 Training and test dataset with XGBoost algorithm

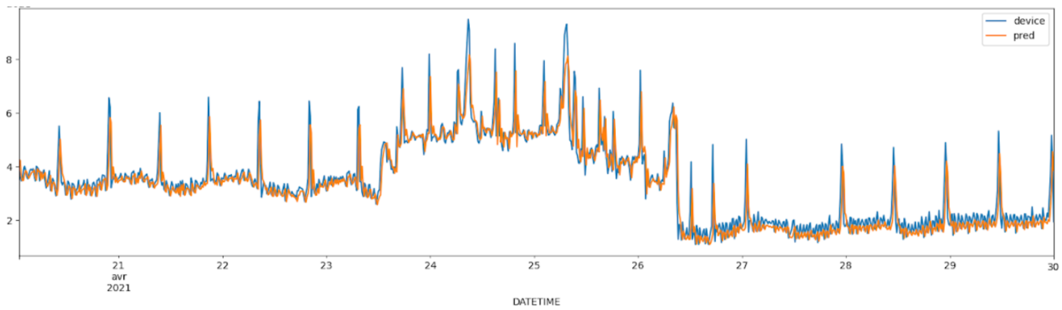

Figure 6.4 Test dataset with LSTM algorithm

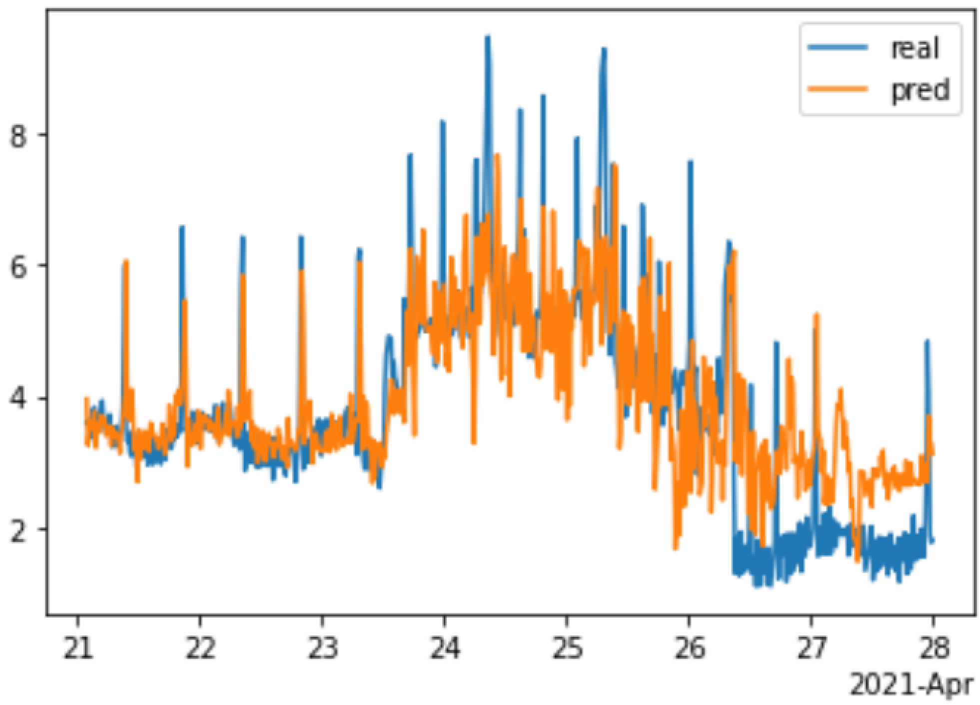

Figure 6.5 Actual value of power vs TFT predictions of power for the last week of April 2021

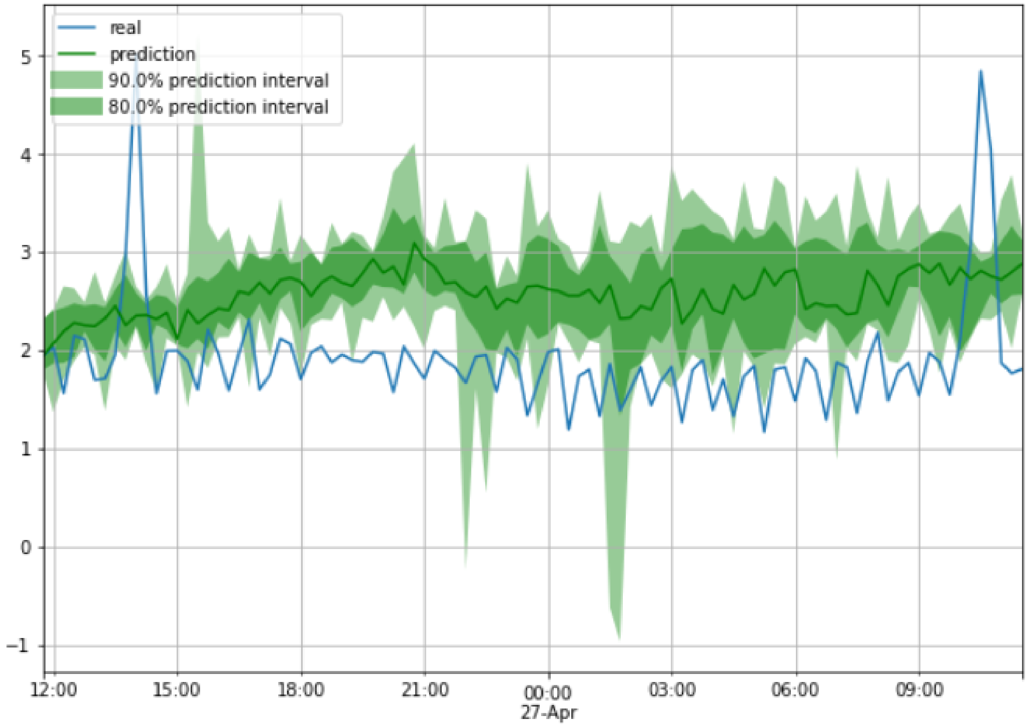

Figure 6.6 *Closer look at actual value of power vs DeepAR predictions of power on the 27th of April*

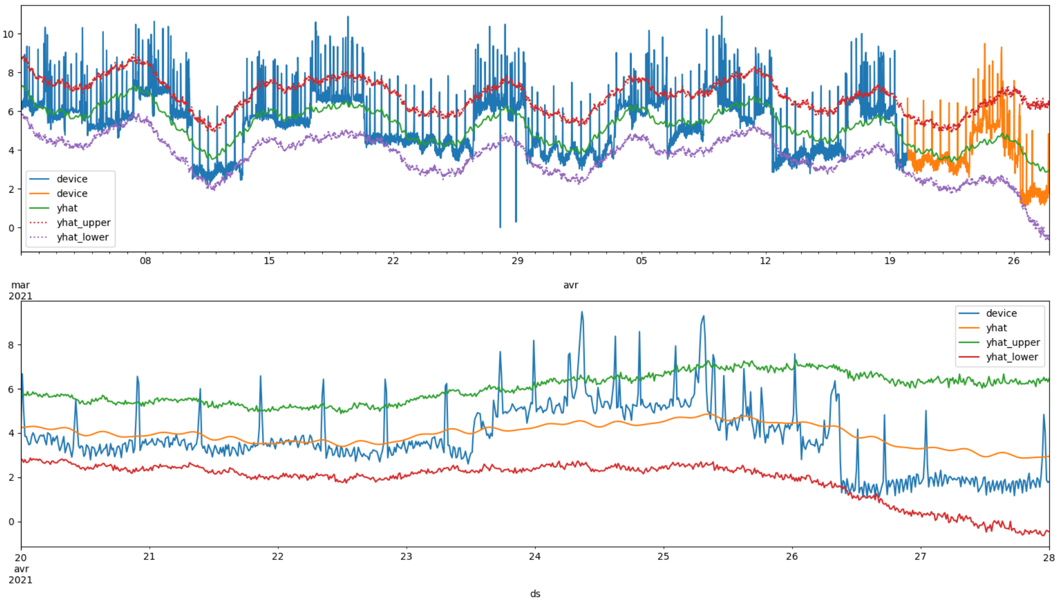

Figure 6.7 *Training and test dataset with Prophet algorithm*

6.3.2 Comparison on several devices

To balance the results, XGBoost and LSTM has been tested on several devices. The algorithm was run over the first 300 devices, but 37 of them have missing value, so the result is over the 263 first devices. In the Figure 6.8, for each device, a new model is trained and the error over the test period is represented. The R2 score is in the top graph, the MSE score in the middle graph and the MAE score in the bottom graph. The result for XGBoost is in blue and the result for LSTM is in orange. We can see that the two algorithms seem equivalent. The Figure 6.9 is the same, but the model is trained with the data of ten successive devices. The model is more generic, but similarities are not considered. Similar devices should be grouped into cluster. For the Figure 6.10, only one model is trained over the 270 first devices. The model is totally generic and can be deployed in the same way on all GED.

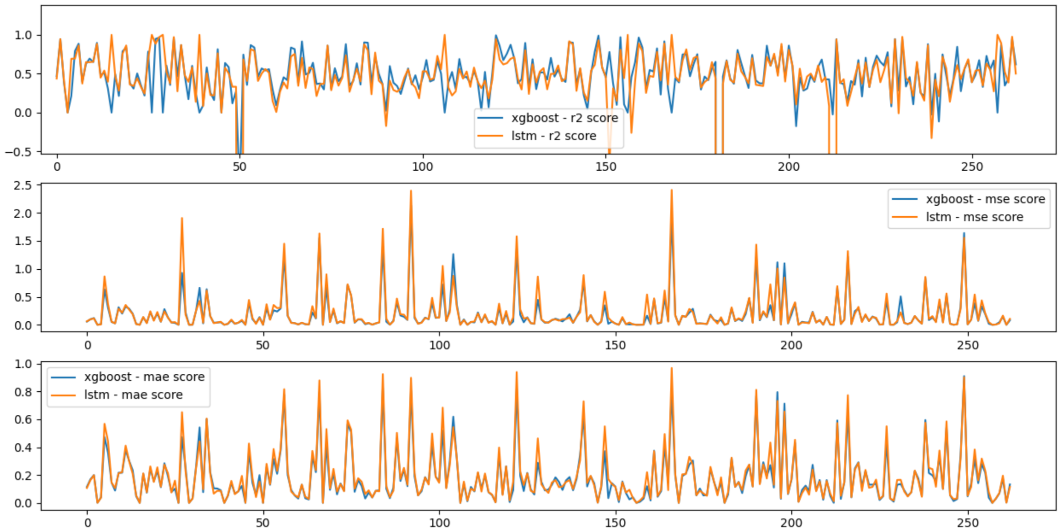

Figure 6.8 Single device model

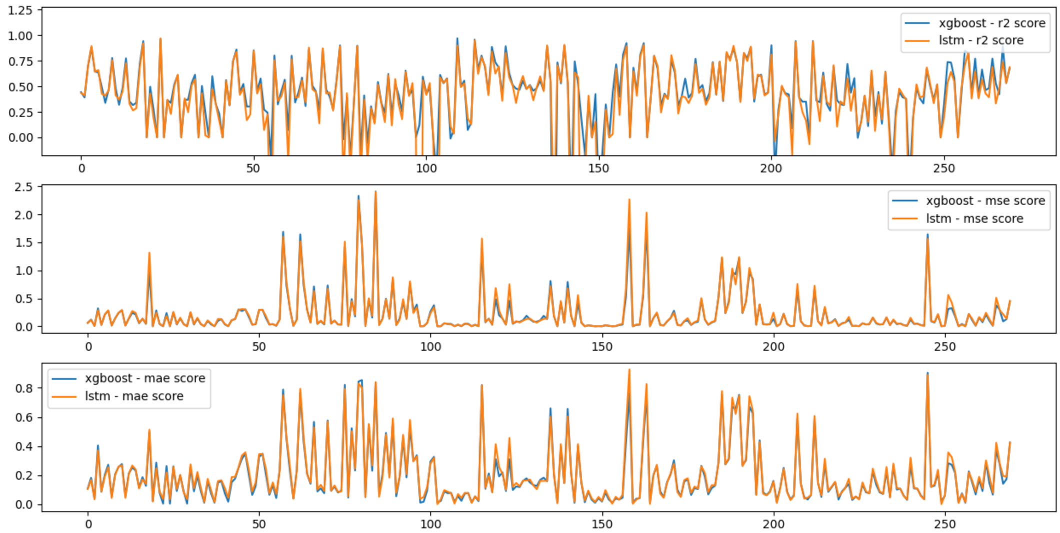

Figure 6.9 Ten devices model

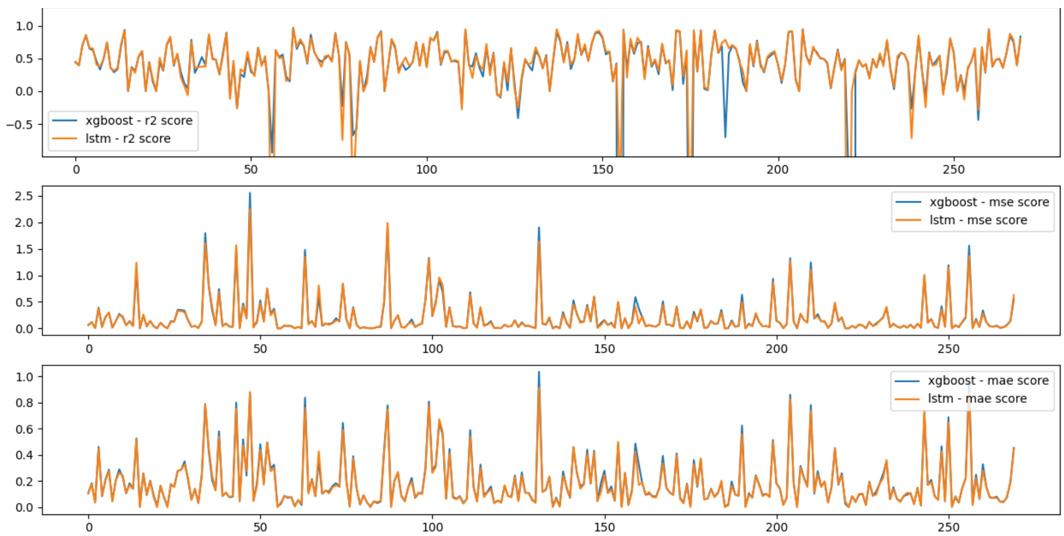

Figure 6.10 One model with 270 devices

The Figure 6.11 compares the performance of the XGBoost algorithm with training on a single device, on a group of ten, and on all 270 devices. The Table 6.5 classify the performance according to the method, with the MSE score. This ranking allows saying that XGBoost is more efficient when trained on a single device. The grouping by ten gives a bad score because the similarities are not considered. A clustering would perhaps improve the result.

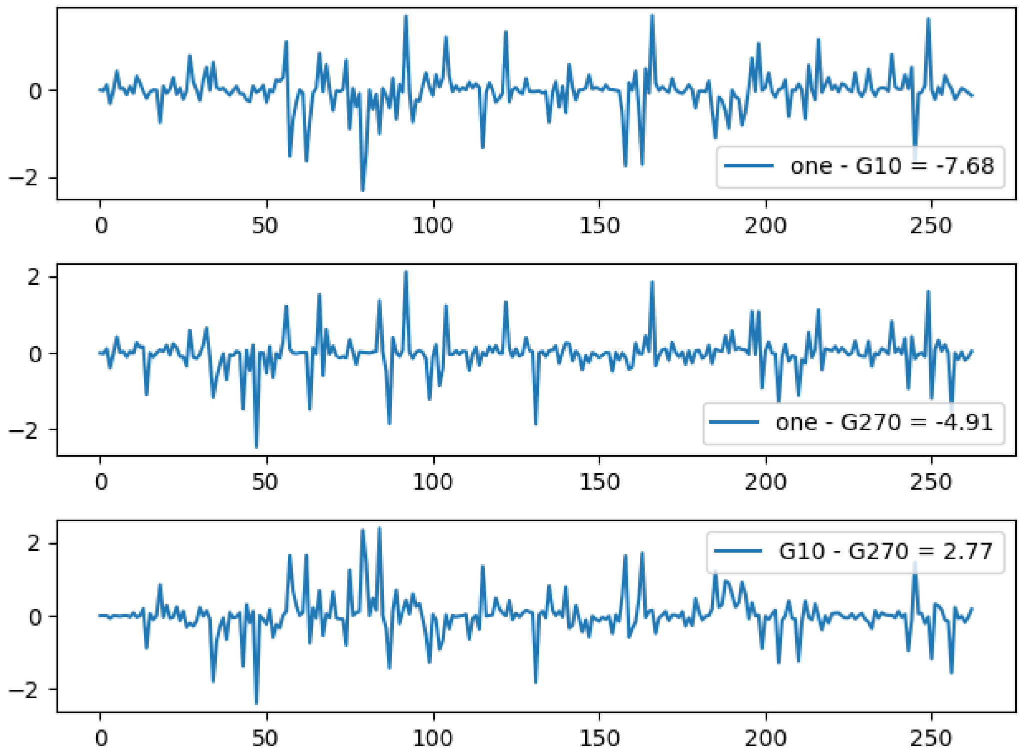

Figure 6.11 XGBoost comparison with MSE score

Biggest error		Smaller error
10 devices for 1 model	**270** devices for 1 model	**1** device for 1 model

Table 6.5 MSE error for XGBoost algorithm according to number of devices for training

The LSTM is compared similarly with the difference in Figure 6.9, and the ranking in Table 6.6. For this algorithm, unlike the other one, with only one model trained on 270 devices, the performance is better. Training on ten devices still gives bad results.

Biggest error		Smaller error
10 devices for 1 model	**1** devices for 1 model	**270** device for 1 model

Table 6.6 MSE error for LSTM algorithm according to number of devices for training

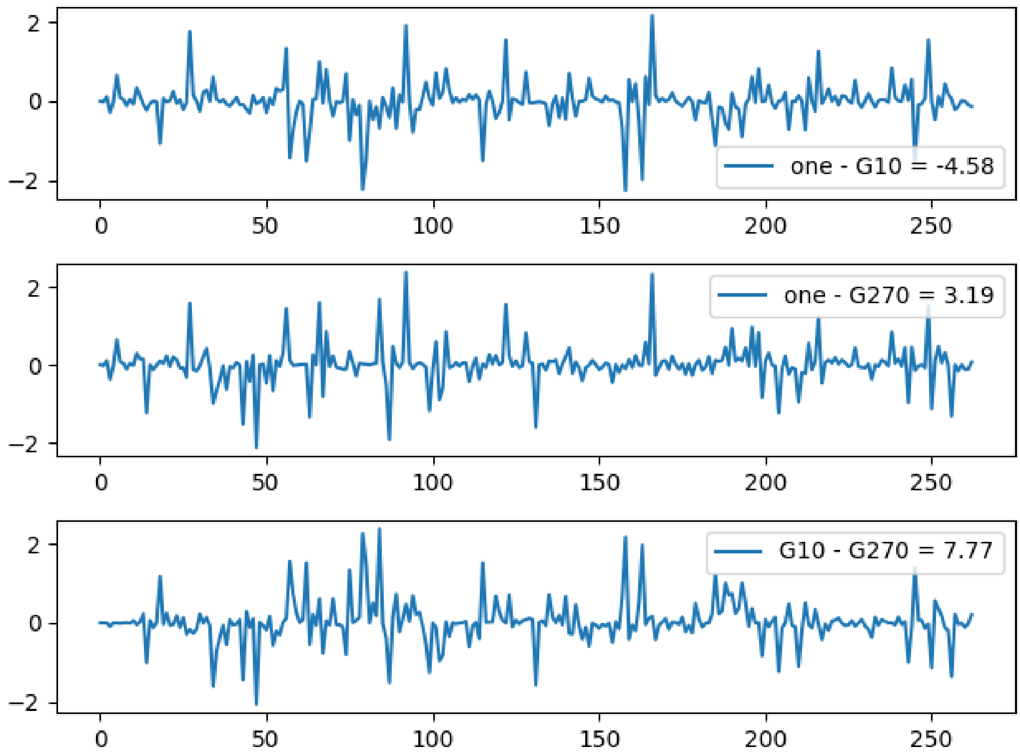

Figure 6.12 LSTM comparison with MSE score

Average MSE (kW)	1 device	10 devices	270 devices
XGBoost	0.194	0.223	0.213
LSTM	0.217	0.234	0.205

Table 6.7 Average MSE for XGBoost and LSTM according to number of devices used for training

The Figure 6.13 compares the LSTM and XGBoost algorithms with the MSE score. We can deduct from the figure that with only one device to train the model, XGBoost has a lower error, but when the number of devices used for training increases, the LSTM algorithm becomes more interesting.

With the LSTM algorithm, it is possible to choose the number of outputs, in other words the number of predictions in the future. The Figure 6.14 compares the results obtained with a single output (the next 15 minutes) or with 96 outputs (the next 24 hours). Obviously, with more value in output, and a more distant horizon, the error is bigger.

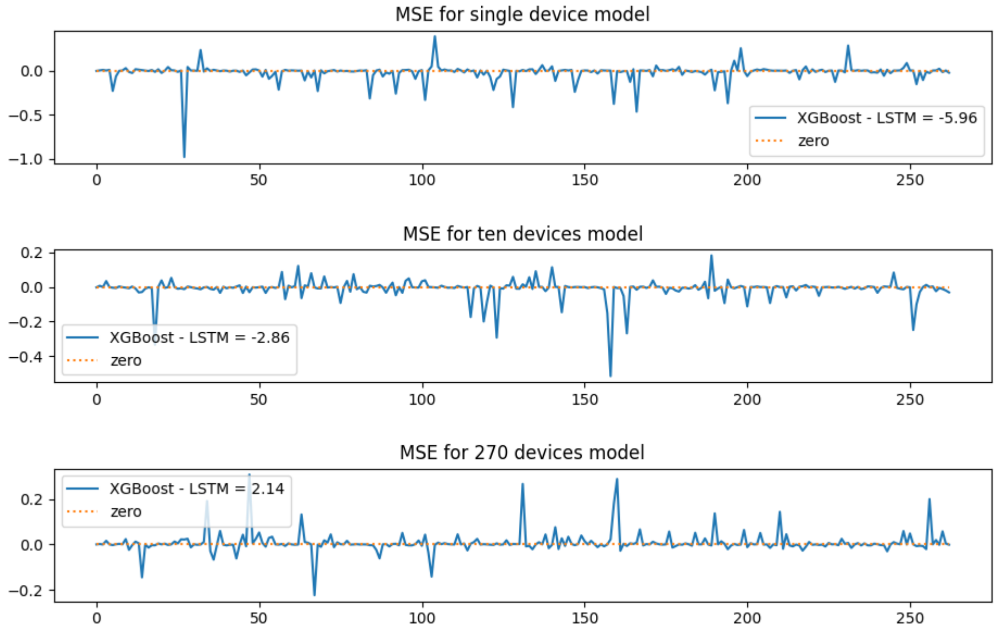

Figure 6.13 Comparison between XGBoost and LSTM with MSE score according to number of device for training

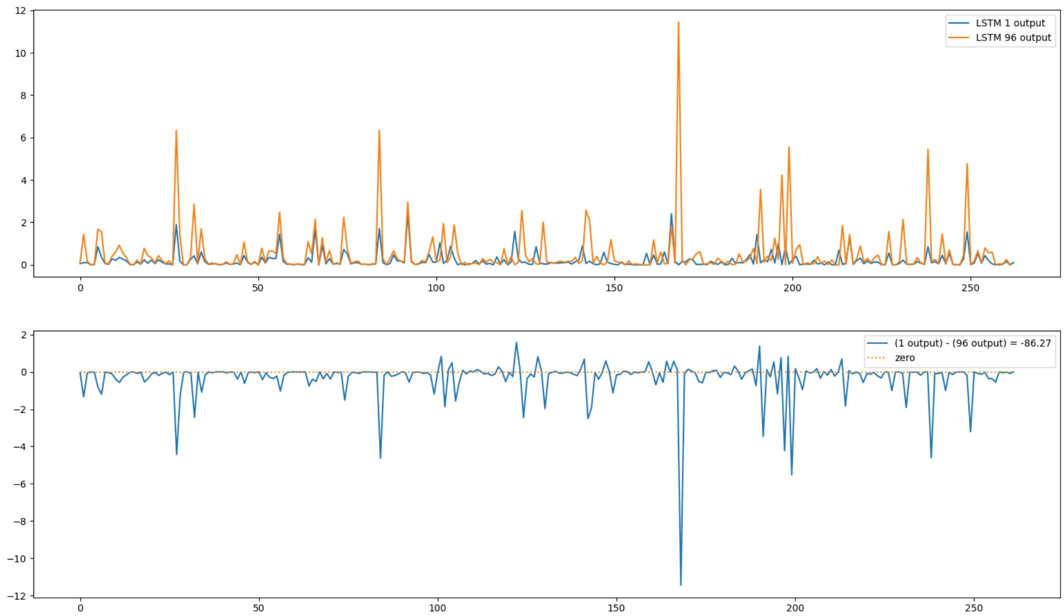

Figure 6.14 Comparison LSTM: 1 and 96 outputs in the first graph. Difference between them in the second

6.3.3 Clustering

As the previous results show, training a model with 10 devices does not give good results. To group the devices, it is then necessary to use a clustering method.

The challenge is to choose the right input parameters for the clustering algorithm. The $pandas.DataFrame.describe()$ method was used, the values obtained in the Table 6.8 are then used with the k-means algorithm. This algorithm aims to group the objects according to the distance with a point which is the center of the cluster.

mean	std	min	10%	
20%	30%	40%	50%	
60%	70%	80%	90%	max

Table 6.8 Describe method output

When using the k-mean algorithm, the most important parameter is the choice of the number of clusters k. For this, the Figure 6.15 helps us to choose. It represents the sum of squared distances between the device and the center of the cluster, with the number of clusters on the x-axis. Then the elbow on the arm is optimal k. The Figure 6.16 represents the number of devices assign to each cluster.

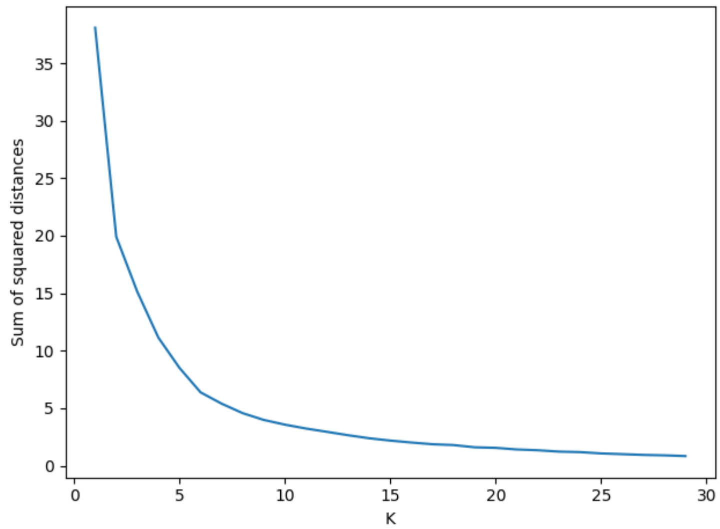

Figure 6.15 Sum of squared distances

Choose about ten cluster seems a good choice. But, as the classes aren't well balanced, and as the clustering method doesn't seem relevant, it will not be used.

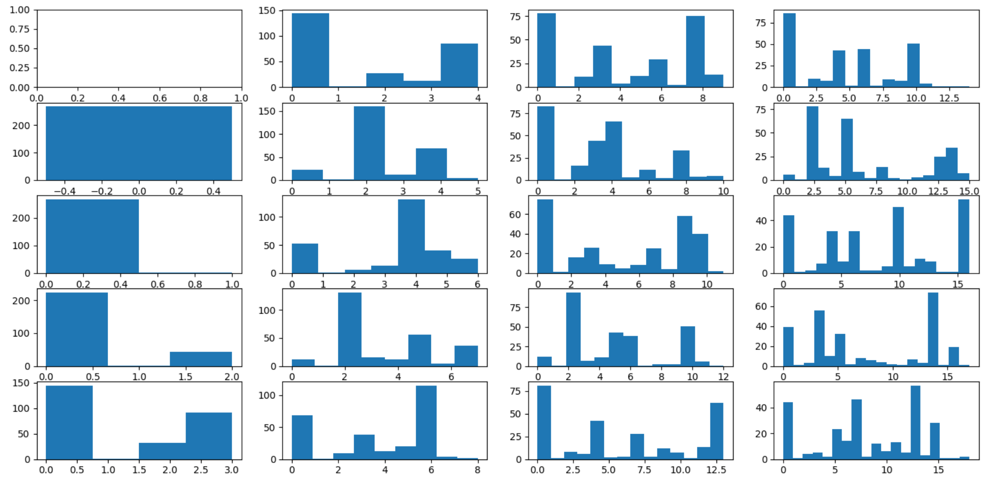

Figure 6.16   Number of device in each cluster histogram

6.4 Conclusion

After describing how the different proposed algorithms work and testing them, we can classify them into two categories based on the execution location. The following points differentiate the algorithms suitable for inference on a GED or on the cloud.

- **Edge devices** XGBoost and LSTM. Light, with good results for a short-term forecast.
- **Cloud** DeepAR, TFT and Prophet. More complex, requiring a larger history, but capable of longer-term prediction.

For the algorithms of the first category, three approaches to build the model are tested, described in the following list.

- A specific and customized model for each GED. For this XGBoost is preferred with good results and low learning time. But this method has the disadvantage of having to generate a model for each GED.
- Build a model with a small set of devices. For this you need a good grouping method. If you don't group similar devices, the results are bad.
- A generic model for all GED. This method requires only one training, but it will be longer because there is more data. The LSTM algorithm gives better results with this approach.

The results presented here are valid for the groupe e dataset which is very heterogeneous, with a time step of 15 minutes. At the time of this report, the data collected by the deployed devices are still insufficient to test the algorithms. In addition, the time step at the GED level is reduced (10 seconds or 1 minute depending on the measurements). These additional data can improve the model but make it more complex.

Ideally, there should be a generic model that can improve in the device with the new measured data or neighbors' data, with transfer learning method.

7 | Conclusions

7.1 Project summary

The electrical energy market is in full mutation. New paradigms are emerging for electrical energy management. Indeed, decentralized electricity production is developing and prosumers are grouping together in micro-grids. This evolution is driven by two main aspects: liberalization and digitalization. In the near future, these two aspects will allow the development of new digital services that will place the citizen, and by extension cities and municipalities, in an active role. The work presented in this report is part of this context. Its ultimate objective is to develop basic digital services for consumption/production prediction at the household (low voltage) and micro grid (medium voltage) levels.

The objective of this project was to deploy and connect CLEMAP grid edge devices in an existing micro-grid infrastructure. Retrieve power consumption data. Connect the CLEMAP devices to an edge-to-cloud platform and deploy forecasting modules.

7.2 Comparison with the initial objectives

In the state of the art part, different micro-grid management solutions with a focus on devices have been compared. Edge-to-cloud infrastructure from several providers have also been compared. The deployment of grid edge devices in a micro-grid has been carried out. The devices are connected to an edge to cloud infrastructure allowing data collection and application deployment. An application to collect measured values and the conservation of these in the cloud for future use as input of machine learning algorithm has been deployed. Algorithms for predicting electricity consumption (on groupe-e dataset) are compared, and intermediate experimental results are presented. These results do not claim to be fully validated, and must be tested on the data measured by the deployed devices. They will be used as a starting point for future research aimed at developing low and medium level energy consumption and production prediction tools. Finally, the deployment of the forecasting modules has not been done because the algorithm has not been chosen yet.

7.3 Encountered difficulties

The first challenge was the duration of the grid edge devices deployment in the Meyrin school with delivery delays, coordination and decision making meetings, and school's network connection times.

The second obstacle is the limited power resource of the CLEMAP devices. It works well as sold, but adding an edge-to-cloud platform, plus a prediction application pushes the limits, and raises availability issues. This project requires either a lighter edge-to-cloud environment or a computer with more resources.

Finally, the possibilities to choose the right prediction algorithm are huge, and it is important to clearly define the criteria. Moreover, the more complex the algorithms are, and the larger the training dataset, the longer the execution times are.

7.4 Future perspectives

The scope of this project is interesting, the subject is already exploited by other competing companies, but there is still room for newcomers. Many innovations are still possible.

The perspectives for this project are large and varied. We can cite the robustness improvement of the GED, the deployment in other micro-grids (apartment buildings ...), and the choice and the improvement of the prediction algorithm in order to have results with little error for a heterogeneous deployment environment.

A | Appendix

A.1 Power in alternating mode

When using Direct Current (DC) the power is obtained simply with the following formula:

$$P = U \cdot I$$

But with Alternative Current (AC) which is used by electric power suppliers, the voltage and current evolve in time and can be out of phase as can be seen in Figure A.2. When the load that uses the power has a capacitive circuit, the voltage lags the current. Conversely, when the circuit is inductive, the current lags the voltage. This is explained by the charging time of a capacitor or a coil which is not instantaneous.

Active power (or real power) is the average power delivered over a given period of time. Noted P, it is expressed in watts (W). For a sinusoidal voltage of rms value U_{eff} and a sinusoidal current of rms value I_{eff}, phase shifted by an angle φ with respect to the voltage, we obtain the formula :

$$P = U_{eff} \cdot I_{eff} \cdot cos\varphi = \frac{U_{max} \cdot I_{max}}{2} \cdot cos\varphi$$

In sinusoidal regime, the ratio between maximum and RMS value is $\sqrt{2}$. $\cos\varphi$ corresponds to the power factor, illustrated in the Figure A.2

The **apparent power** in alternating mode is the product of the effective value of the electric voltage at the terminals of the dipole by the effective value of the electric current passing through this dipole

$$S = U_{eff} \cdot I_{eff} = \frac{U_{max} \cdot I_{max}}{2}$$

In sinusoidal mode, the **reactive power** is the imaginary part of the complex apparent power. It is noted Q, is expressed in reactive voltamper (var). It corresponds to lost energy, dissipated in heat.

$$Q = U_{eff} \cdot I_{eff} \cdot sin\varphi = \frac{U_{max} \cdot I_{max}}{2} \cdot sin\varphi$$

[1] Source: http://fr.wikipedia.org/wiki/Puissance_en_régime_alternatif

[2] Source : http://fr.wikipedia.org/wiki/Facteur_de_puissance

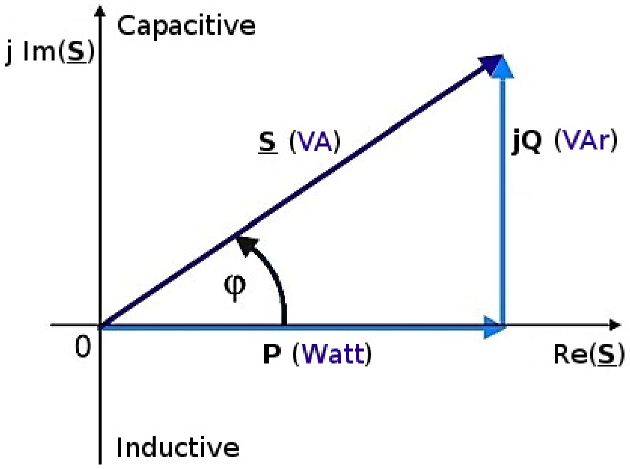

Figure A.1 Power vector [1]

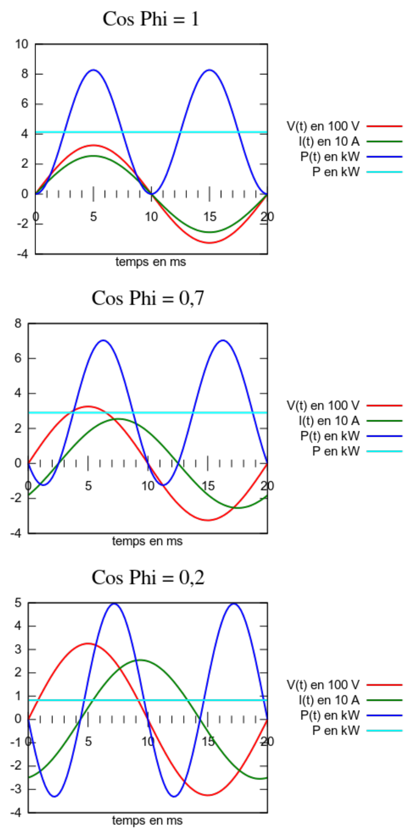

Figure A.2 Power factor [2]

References

[1] Swiss Federal Office Of Energy Sfoe. *Energy Strategy 2050*. Feb. 2022. URL: https://www.bfe.admin.ch/bfe/en/home/policy/energy-strategy-2050.html.

[2] *IEEE Standard for the Specification of Microgrid Controllers*. Apr. 2018. DOI: 10.1109/IEEESTD.2018.8340204.

[3] Sma Solar Technology Ag. *Wechselrichter, Photovoltaik- & Solartechnik*. 2022. URL: https://www.sma.de/en.html.

[4] *EEBUS | Make the world speak energy - EEBus Initiative e.V.* Dec. 2020. URL: https://www.eebus.org.

[5] Dr Andreas Beer and Dr Maurus Bachmann. *Limitation de la puissance d'injection photovoltaïque | Association Smart Grid Suisse*. Aug. 2020. URL: https://smartgrid-schweiz.ch/fr/2020/04/16/pv-einspeiselimitierung.

[6] Sma Solar Technology Ag. *Sunny Home Manager 2.0*. Feb. 2022. URL: https://www.sma.de/fr/produits/surveillance-controle/sunny-home-manager-20.html.

[7] *Home*. 2021. URL: https://hivepower.tech.

[8] *The Project – LIC*. Mar. 2022. URL: https://lic.energy/the-project.

[9] *Strato Pi - The Industrial Raspberry Pi - Sfera Labs*. Feb. 2022. URL: https://www.sferalabs.cc/strato-pi.

[10] *Intel® NUC Products*. Mar. 2022. URL: https://www.intel.com/content/www/us/en/products/details/nuc.html.

[11] *NEMoGrid – NEMoGrid ERA-NET SG+ Project*. 2022. URL: https://nemogrid.eu.

[12] Smart Grids Plus. *ERA-Net Smart Energy Systems: APPROVED PROJECTS AND THEIR OUTCOMES*. Feb. 2022. URL: https://www.eranet-smartenergysystems.eu/Projects.

[13] Cordis. europa. eu Cordis. "Pro-sumer AwaRe, Transactive Markets for Valorization of Distributed flexibilITY enabled by Smart Energy Contracts". In: *CORDIS | European Commission* (Sept. 2019). URL: https://cordis.europa.eu/project/id/864319.

[14] Cordis. europa. eu Cordis. "deMonstration of smArt and flExible solutions for a decarboniSed energy future in Mayotte and other European islAnds". In: *CORDIS | European Commission* (Sept. 2020). URL: https://cordis.europa.eu/project/id/957843.

[15] *With CLEMAP into the energy future*. Feb. 2022. URL: https://en.clemap.ch.

[16] *Modicon Modbus Protocol Reference Guide*. Feb. 2022. URL: https://www.modbus.org/docs/PI_MBUS_300.pdf.

[17] Ltd. *Raspberry Pi*. Feb. 2022. URL: https://www.raspberrypi.com.

[18] *PowerID: Community Energienetzwerke mit Prosumenten-Fokus (Projekt «Quartierstrom») - Textes*. Feb. 2021. URL: https://www.aramis.admin.ch/Texte/?ProjectID=40844.

[19] *SmartPi - nD-enerserve GmbH*. [Online; accessed 16. Feb. 2022]. Feb. 2022. URL: https://www.enerserve.eu/en/smartpi.html.

[20] *Revolution Pi - THE ONLY REAL industrial Raspberry Pi*. Jan. 2022. URL: https://revolutionpi.com.

[21] Ltd. "Operating system images – Raspberry Pi". In: *Raspberry Pi* (Jan. 2021). URL: https://www.raspberrypi.com/software/operating-systems.

[22] *Intelligence à la périphérie IoT – AWS IoT Greengrass – Amazon Web Services*. Feb. 2022. URL: https://aws.amazon.com/fr/greengrass.

[23] *IoT Edge | Cloud Intelligence | Microsoft Azure*. Mar. 2022. URL: https://azure.microsoft.com/fr-fr/services/iot-edge/#iotedge-overview.

[24] *Google Cloud IoT – Services IoT entièrement gérés | Google Cloud*. Dec. 2021. URL: https://cloud.google.com/solutions/iot.

[25] *Coral*. Mar. 2022. URL: https://coral.ai.

[26] *balena - The complete IoT fleet management platform*. Mar. 2022. URL: https://www.balena.io.

[27] SixSq S. A. *Enable your edge with our management platform as a service*. Feb. 2022. URL: https://nuvla.io.

[28] *Smart City Software Platform - CityLinx™ control management system*. June 2021. URL: https://beezeelinx.com.

[29] *Edge Impulse*. Feb. 2022. URL: https://www.edgeimpulse.com.

[30] *Link IoT Edge: Edge Capabilities - Alibaba Cloud*. Feb. 2022. URL: https://www.alibabacloud.com/product/linkiotedge.

[31] *IBM Edge Application Manager - Overview*. Aug. 2021. URL: https://www.ibm.com/cloud/edge-application-manager.

[32] *Smart and Widely-distributed Appliances for Renewable energy Management | HEPIA inIT LSDS*. 2022. URL: https://lsds.hesge.ch/smart-and-widely-distributed-appliances-for-renewable-energy-management.

[33] *Les Vergers | Ecoquartier Meyrin Les Vergers*. 2022. URL: https://www.lesvergers-meyrin.ch/ecoquartier/les-vergers.

[34] *Minergie-A - MINERGIE Schweiz*. Mar. 2022. URL: https://www.minergie.ch/fr/certifier/minergie-a.

[35] *digitaL frAmework for SmArt Grid and reNewable Energie | HEPIA inIT LSDS*. Mar. 2022. URL: https://lsds.hesge.ch/digital-framework-for-smart-grid-and-renewable-energie.

[36] *NuvlaBox Engine Architecture*. 2022. URL: https://docs.nuvla.io/nuvlabox/nuvlabox-engine/v2/architecture.

[37] *3.3. Metrics and scoring: quantifying the quality of predictions*. 2022. URL: https://scikit-learn.org/stable/modules/model_evaluation.html.

[38] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *arXiv* (Mar. 2016). DOI: 10.1145/2939672.2939785. eprint: 1603.02754.

[39] *DeepAR Forecasting Algorithm - Amazon SageMaker*. Feb. 2022. URL: https://docs.aws.amazon.com/sagemaker/latest/dg/deepar.html.

[40] *Project Jupyter*. Feb. 2022. URL: https://jupyter.org.

[41] David Salinas et al. "DeepAR: Probabilistic forecasting with autoregressive recurrent networks". In: *International Journal of Forecasting* 36.3 (July 2020), pp. 1181–1191. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2019.07.001.

[42] Bryan Lim et al. "Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting". In: *arXiv* (Dec. 2019). DOI: 10.48550/arXiv.1912.09363. eprint: 1912.09363.

[43] Sean J. Taylor and Benjamin Letham. "Forecasting at scale". In: *PeerJ Preprints* (Sept. 2017). ISSN: 2167-9843. DOI: 10.7287/peerj.preprints.3190v2.

[44] *Quick Start*. Feb. 2022. URL: https://facebook.github.io/prophet/docs/quick_start.html.

[45] *Groupe E | Experts en solutions énergétiques globales*. Mar. 2022. URL: https://www.groupe-e.ch/fr.

[46] *Grid levels*. Mar. 2022. URL: https://www.swissgrid.ch/en/home/operation/power-grid/grid-levels.html.

[47] *RS 734.71 - Ordonnance du 14 mars 2008 sur l'approvisionnement en électricité (OApEl)*. 2022. URL: https://www.fedlex.admin.ch/eli/cc/2008/226/fr#art_8_a.

Glossary

A Ampere. 27

AC Alternative Current. 29, 65

ADF Augmented Dickey-Fuller. 38

AI artificial intelligence. 15, 35

AMQP Advanced Message Queuing Protocol. 19

ANN Artificial Neural Networks. 40

API application programming interface. 16

AWS Amazon Web Services. 18, 41

CLI command line interface. 16, 17

CVE Common Vulnerabilities and Exposures. 32

DC Direct Current. 65

DeepAR Deep Auto-Regressive. 41

DNN Deep Neural Network. 42

EC2 Elastic Compute Cloud. 41

ERA-Net European Research Area Network. 9

EV Electric Vehicle. 2, 6, 10

FN False Negative. 36

FP False Positive. 36

GCP Google Cloud Platform. 15, 20

GED grid edge device. 2, 6–11, 13, 15, 25, 27, 29, 33, 45, 47, 55, 61–64

GPIO General Purpose Input/Output. 28

GUI graphical user interface. 17

Hz Hertz. 28

IDE integrated development environment. 20

IIoT Industrial Internet of Things. 11

IoT Internet of Things. 15, 20, 23

IP Internet Protocol. 11

KPSS Kwiatkowski-Phillips-Schmidt-Shin. 38

LASAGNE digitaL frAmework for SmArt Grid and reNewable Energie. 25, 27

LIC Lugaggia Innovation Community. 9

LSTM Long Short-Term Memory. 35, 40, 41, 51, 52, 55, 57, 58, 61

MAE Mean Absolute Error. 37, 51, 55

MAESHA deMonstration of smArt and flExible solutions for a decarboniSed energy future in Mayotte and other European islAnds. 10

ML Machine Learning. 3, 15, 20, 41

MLM machine learning model. 15–17, 23

MQTT Message Queuing Telemetry Transport. ix, 17, 19, 29, 31

MSE Mean Squared Error. 37, 55, 56, 58

NBE NuvlaBox Engine. 31

NEMoGrid New Energy business Models in the distribution Grid. 9

NN Neural Networks. 39–41

P2P Peer-to-Peer. 11

PARITY Pro-sumer AwaRe, Transactive Markets for Valorization of Distributed flexibilITY enabled by Smart Energy Contracts. 10

PV PhotoVoltaics. 6, 8

RMSE Root Mean Squared Error. 37, 51

RNN Recurrent Neural Networks. 40, 41

RTC Real Time Clock. 12

SDK software development kit. 16

SWARM Smart and Widely-distributed Appliances for Renewable energy Management. 24, 25, 30, 47

TFT Temporal Fusion Transformers. 42, 51, 61

TN True Negative. 36

TP True Positive. 36

TPU Tensor Processing Unit. 20

UPS Uninterruptible Power Supply. 12

V Volt. 27

VA Volt Ampere. 27

VAr reactive Volt Ampere. 27

VPN Virtual Private Network. 11, 32

W Watt. 27

XGBoost eXtreme Gradient Boosting. 39