TECHNISCHE UNIVERSITÄT WIEN

Institut für Computertechnik
Institute of Computer Technology

A MASTER THESIS ON

Pattern Recognition in a heterogeneous Smart Grid environment

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

Diplom-Ingenieur
(Equivalent to Master of Science)

in

Energie und Automatisierungstechnik (UE 066 506)

by

Matthias Bittner

01425398

Supervisor(s):

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Thilo Sauter

Univ.Ass. Dipl.-Ing. Daniel Hauer, BSc

Vienna, Austria

June 2021

Abstract

Considering a Smart Grid and just observing the sampled grid measurements is an old-fashioned and outdated way of looking at this highly dynamical and heterogeneous system. There is a strong need of involving the environmental (e.g., weather, seasonal behaviour) and heterogeneous (e.g., diverse energy sources and consumers) influences into their analysis and optimization.

This thesis is therefore starting at a very abstract viewpoint of such a Smart Grid and proposes:
a pipeline for extracting patterns and a design cycle for developing Machine Learning concepts.

The pattern extraction pipeline provides methods and concepts for extracting patterns related to the environmental and heterogeneous influences. This step of revealing and extracting patterns is achieved by applying this pipeline on historical data of an existing testbed in Aspern Vienna, Austria.

The second part of this thesis is then focused on proposing a Machine Learning design cycle, which provides a general methodology for developing Machine Learning concepts based on the extracted patterns. This results in concepts for power consumption forecasting based on environmental data, system state clustering and rare event detection. The overall aim of all these concepts is to optimize the functionality, reliability and efficiency of the modern Smart Grids.

Kurzfassung

Bei einem Smart Grid nur die abgetasteten Netzmesswerte zu beobachten, ist eine altmodische und veraltete Art, solch ein hochdynamisches und heterogenes System zu betrachten. Vielmehr ist es notwendig, die Einflüsse der Umwelt (z.B. Wetter, saisonales Verhalten) und der Heterogenität (z.B. unterschiedliche Energiequellen und Verbraucher) in die Analyse und Optimierung mit einzubeziehen. Diese Arbeit geht daher von einer sehr abstrakten Sichtweise eines Smart Grids aus und stellt eine Pipeline für das Extrahieren von Mustern und einen Machine Learning Design Cycle vor.

Die Pipeline legt Methoden und Konzepte zur Extraktion von Mustern in Bezug auf die Umwelt und die heterogenen Einflüsse dar. Dieser Schritt des Erkennens und Extrahierens von Mustern wird durch die Anwendung dieser Pipeline auf historische Daten eines bestehenden Testbeds in Aspern Wien, Österreich, erreicht.

Der zweite Teil dieser Arbeit konzentriert sich auf den Machine Learning Design Cycle, der eine allgemeine Methodik für die Entwicklung von Machine Learning Konzepten basierend auf den extrahierten Mustern bereitstellt. Dies resultiert in Konzepten für die Vorhersage des Energieverbrauchs auf Basis von Umwelteinflüssen, das Clustering von Systemzuständen und die Erkennung von seltenen Events. Das übergeordnete Ziel all dieser Konzepte ist es, die Funktionalität, Zuverlässigkeit und Effizienz solch moderner Smart Grids zu optimieren.

Preface

Parts of chapter 6 are related to the paper [1]:

Daniel Hauer, Matthias Bittner, S. Cejka, R. Mosshammer, F. Kintzler, T. Leopold, S. Wilker,

"Re-enacting rare multi-modal real-world grid events to generate ML training data sets",

IEEE International Symposium on Industrial Electronics, 2021.

This paper is already accepted and will be published by the end of June 2021.

Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt wurde. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.
Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Copyright Statement

I, Matthias Bittner, hereby declare that this thesis is my own original work and, to the best of my knowledge and belief, it does not:

- Breach copyright or other intellectual property rights of a third party.
- Contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
- Contain material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.
- Contain substantial portions of third party copyright material, including but not limited to charts, diagrams, graphs, photographs or maps, or in instances where it does, I have obtained permission to use such material and allow it to be made accessible worldwide via the Internet.

Signature: _____

Vienna, Austria, June 2021 Matthias Bittner

Acknowledgment

You can find the right direction and reach your goals only, if you know where you are now and when you know how things are around you.
My family and friends always provided me a compass to keep track of my values, wishes and goals. Thank you!

Contents

List of Tables

List of Figures

Acronyms

CNN Convolutional Neural Network. 8

DL Deep Learning. 12, 19, 48, 51

FNN Feed-forward Neural Network. 8

LSTM Long Short-Term Memory. 8, 51

ML Machine Learning. ix, 3, 4, 7, 8, 9, 10, 11, 12, 16, 18, 19, 30, 32, 36, 40, 41, 42, 49, 50

NN Neural Network. 11, 58

RNN Recurrent Neural Network. 8

Chapter 1

Introduction

Imagine observing the world from space without any prior knowledge about its structure, inhabitants or any bias you can imagine. First things which might be visible at sunlight are areas preserved by nature and taking a closer look one will recognize areas which are influenced by human kind. Giving our observation a second try without any sunlight hitting the surface, we will notice, that the perception of the beautiful nature will be kept in the dark and the cities where we humans are living will shine bright. This bright and also beautiful viewpoint of the world is caused by electricity and its exploration has fundamentally changed the life of human beings.

Ever since this exploration of electricity, humans are continuously trying to improve its usage, storage, distribution and generation. While the industrial revolutions have pushed forward the research and applications of electricity, it is currently known as a normal and integral part of our lives.

Since we are now living in a modern and complex world, where resources are slowly depleting, old fashioned grid systems need to be renewed, optimized and monitored in order to ensure that the rapidly increasing demand of electrical energy can be ensured. These problems are nowadays solved with Smart Grid concepts, which should provide frameworks to prevent scenarios like e.g., power outages, system failures or energy shortages. These modern Smart Grid concepts are following various new approaches but are still under development. Additionally they do face some major challenges regarding the actual analysis and interpretation of all the monitored information.

This thesis is therefore trying to tackle some of the open research challenges and especially taking into account a more abstract viewpoint of the Smart Grid. This viewpoint might not be out of space, but it will consider a heterogeneous Smart Grid environment and investigate the influence of the environment onto the grid to extract patterns. Once these patterns are defined one can make use of this knowledge to actually design machine learning applications for detecting these patterns or for generating even more information based on the knowledge of the patterns.

1.1 The heterogeneous Smart Grid environment

When observing a Smart Grid as it is currently developed, then we consider a heterogeneous structure of energy consumers (e.g., industry buildings, electric cars, private/public buildings), energy storage systems and energy producers (e.g., photo-voltaic systems, wind farms). This so called *prosumers*, are simultaneously influencing the grid and show that the system has to deal with bidirectional and sometimes unpredictable load flows. Environmental factors like weather, seasonal information, socio-economic behavior (residents, social media) or economic properties (energy policies, dynamic market prices) can be seen as additional influence to the Smart Grid. Especially the behaviour of us humans is of high interest. Social Media, politics or even a pandemic can have a huge impact on the way how we are interacting with each other and consuming energy. Especially thinking of events which are advertised and published on social media, it is not a hidden secret, that these can have a enormous range in a short time, which is therefore also influencing the power consumption in the areas where these events are promoted. Figure 1.1 shows such a possible heterogeneous Smart Grid configuration, which is build with the Smart Grid co-simulation tool Bifrost[1].

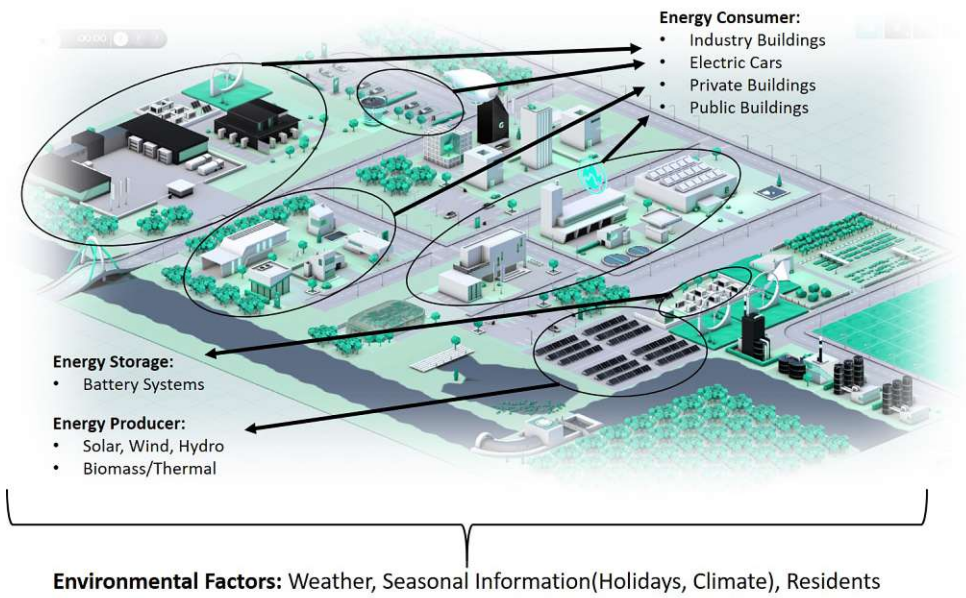

Figure 1.1: Heterogeneous Smart Grid environment in Bifrost

A Smart Grid, as an intelligent systems itself, is now a source of both energy and information. The information which is collected from the grid consists of sensor data about various processes like, electricity generation, transmission, distribution and consumption. This data is most of the time measured with grid monitoring devices and therefore includes timeseries and electrical information about distribution stations, distribution switch stations and electricity meters.

[1] https://bifrost.siemens.com/

The enormous amount of internal data about the physical state of the Smart Grid, in combination with the external heterogeneous influence, can now be used to provide information about the security, stability, reliability and efficiency. In the literature, this research area is also referred to big data analytics in Smart Grids and one of the main challenges is still to systematically process and analyse the already existing and heterogeneous data in order to actually be able to generate knowledge [2].
The investigated heterogeneous Smart Grid environment in this work is related to the testbed Aspern[1] in Vienna, Austria. This grid consist of over 12 low-voltage distribution substations and is basically representing the behaviour of a small city. The availability of the grid plan, the knowledge about the different consumers and producers in combination with the historical data about the grid measurements and the corresponding environmental influence (e.g., temperature and solar radiation) form an ideal starting point for investigating and finding patterns which are related to this heterogeneous environment. A detailed explanation of the testbed and the available historical data is provided in chapter 4.

1.2 Current challenges in heterogeneous Smart Grids

Since pattern recognition and big data analytics in Smart Grids is a comprehensive and complicated field, one will agree that the amount of challenges which are occurring when looking through the literature [2–5] will form a long list, but some of them are listed below.

- **Data quality** - Until now you might have read the word *data* multiple times in this work, but at the same time there was no direct definition about what exact data is meant. So one major challenge is the lack of a structured way to take into account all the different sources of data.
- **Data visualization** - Most of the existing Smart Grid big data analytics schemes do not incorporate visualization of the data and extracted knwoledge as an integral part. Since the goal of Machine Learning (ML) and data analysis is to extract patterns which should provide information about the real-time situation it is very important to integrate the visualized knowledge into the design process of Smart Grid frameworks.
- **Heterogeneous influence / knowledge interpretations** - Without the extraction of information or knowledge, all the collected data holds no value. In most cases the generated heterogeneous data is wasted without extracting potentially useful information and knowledge because of the lack of established mechanisms and standards. Additionally most of the data analytic approaches exploit sampling methods that are efficient in terms of time complexity but neglect a large part of data that may contain important patterns which are not represented by the sam-

[2]https://www.ascr.at

ples. Also taking into account the heterogeneous sources of the data (e.g., weather, social media, traffic, holidays) should be considered in the Smart Grid framework in order to uncover hidden information.

- **Lack of sharing data** - Most of the generated data in a Smart Grid can be considered as confidential or related with privacy issues. This makes the process of sharing the data inbetween this research field very complicated and almost not possible. That's an issue when trying to design a standard data format or benchmarking the developed ML algorithms on different data sets.
- **Lack of deployment** - Most of the Machine Learning techniques are assuming a fixed training model on a static data set. These assumptions do not maintain when trying to deploy the methods in the real-time application since the data changes and evolves over time. The trend should be to bring the analytics closer to the edge devices and make use of adaptive ML algorithms. But not only the technological issues are limiting the deployment. In lots of cases business units are still waiting to see convincing, beneficial, economic results before they are willing to make investments for deploying the applications.

Considering all these challenges but especially focusing on investigating how the heterogeneous Smart Grid is influenced by its inherent structure and the environment, this work should provide methods and approaches to contribute to some of the open research challenges.
Taking into account a heterogeneous Smart Grid testbed with the availability of historical grid measurements, but without any other further specifications about the external influences we do face these above challenges and are defining the following research questions.

How can data analysis and machine learning be used to extract patterns and information from the historical data of an heterogeneous Smart Grid environment?

- How can we define the heterogeneous environment and the corresponding data sets, especially considering easy to acquire data?
- Which methods are suitable for extracting patterns?
- How can the extracted knowledge be prepared or visualized in order to be useful?
- Taking into account the application in real Smart Grid devices, which patterns or events can be detected and which Machine Learning concepts are most suitable for this purpose?

1.3 Outline

Chapter 2 provides an overview about current Smart Grid frameworks and commonly used Machine Learning concepts. Chapter 3 is introducing the used methodologies in order to tackle the problem of Pattern Recognition and Machine Learning in a heterogeneous Smart Grid environment.
Applying these methodologies will form the main part of this thesis, whereas in chapter 4 we introduce the underlying heterogeneous Smart Grid environment and its corresponding data-sets in a way to be able to perform data analysis for extracting patterns and knowledge. Once the data analysis is revealing patterns and behaviours connected to the heterogeneous influence we directly step towards chapter 5 where we propose Machine Learning concepts to solve some of the prior extracted problems. Since most of the modern Machine Learning approaches do face the need of high quality data chapter 6 is used to propose a novel concept for simulating Machine Learning training data to detect rare real-world grid events.
Finally chapter 7 is presenting the major conclusions and results. Additionally we will give some outlook and attempts to also address the still remaining research challenges.

Chapter 2

State of the Art

”A Smart Grid uses sensing, embedded processing and digital communications to enable the electricity grid to be observable (able to be measured and visualised), controllable (able to be manipulated and optimised), automated (able to adapt and self-heal), fully integrated (fully interoperable with existing systems and with the capacity to incorporate a diverse set of energy sources).” This statement of the Department of Energy and Climate Change, UK, is one possible definition of a Smart Grid, but as one can already assume also not the only one. In fact, there will be no exact definition of a Smart Grid [6], but it is obvious that the term Smart Grid describes various technologies and approaches which try to transfer the environmentally extravagant conventional grid into a more reliable, flexible and sustainable system [7].

If we look at Figure 2.1, we can see that the Smart Grid has become very popular worldwide since the beginning of 2009, whereby the Smart Grid also plays an essential role in Smart Cities [8] (whose increase in popularity only began a few years later). Since this shift in 2009, where obviously politics and research have recognised the necessity of restructuring the electrical grid, Smart Grid and Smart City concepts are being developed and implemented in testbeds [1] successfully [9].

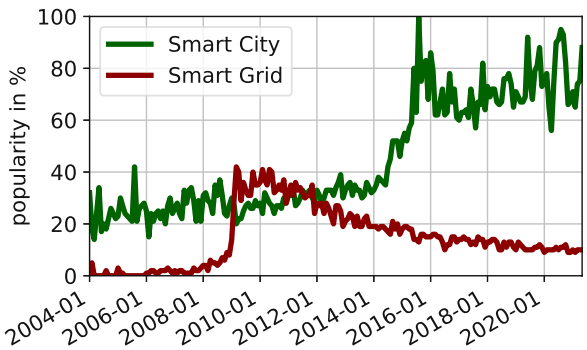

Figure 2.1: Popularity of Smart Cities and Smart Grids (Google Trends 2004-2020)

[1] https://www.smartgrids.at, https://www.ascr.at

Modern Smart Grids are now a source of energy and information. Considering this heterogeneous structure we are facing a situation where we actually need to process and analyse its generated data. It is necessary to uncover hidden behaviour, patterns and to detect safety critical events. Pattern Recognition and Machine Learning are state of the art methods for solving such problems. The next two sections will therefore address state of the art examples for Pattern Recognition in the Smart Grid domain and their underlying Machine Learning concepts.

2.1 Pattern Recognition and Machine Learning in Smart Grids

Integrating Internet of Things (IoT) devices into the grid infrastructure is the first step towards the Smart Grid and its inherent need for data analysis. This focus on implementing the information and communications technology infrastructure to collect data decentralized and to process it centrally, led to a point where the processing and analysis of this enormous amount of data is possible and necessary. Looking at Figure 2.2 one will see the evolution of the interest in big data and Machine Learning considering all research fields (Google Trends 2004-2020). Connecting this to the big data characteristics: volume, velocity, variety and value [5] it can be recognized that the increase in volume and velocity has shifted the interest towards Machine Learning. It seems that the first step was to build the methods and mechanisms to collect data and then the interest was shifted to the analysis of this data with the help of Machine Learning.

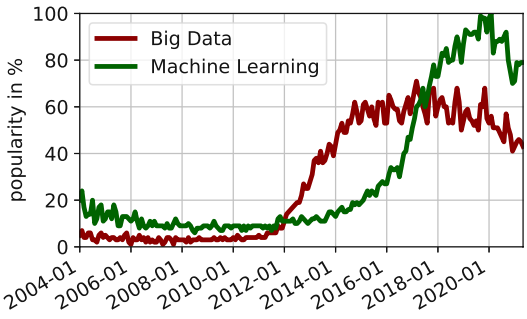

Figure 2.2: Popularity of ML and big data

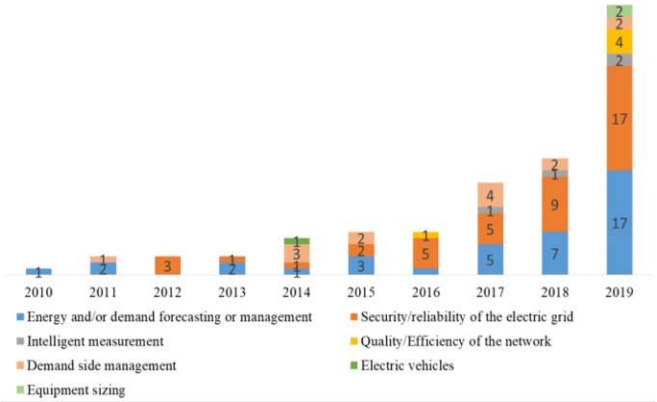

Figure 2.3: Research trends ML and Smart Grid

Regarding the application of ML in Smart Grids, most of the trending topics are related to load/demand forecasting, security and reliability of the grid, cyber security, power system analysis and control, renewable energy forecasting, predictive maintenance/condition based maintenance, power quality monitoring, prediction and detection of faults, load forecasting and load profiling [2, 10]. A quantitative descriptive analysis shows that 72 % of the Smart Grid related ML research was published during the

period from 2010 to 2019 [11]. If one compares Figure 2.2 and 2.3, it is recognisable that the increasing *global* interest in ML, did also result in an increasing interest of ML applications in Smart Grids. An additional interesting insight from Figure 2.3 is that the majority of publications are dealing with energy and demand forecasting (37%) or security/reliability of the grid (39%).

While all these topics aim for different goals (e.g., cost optimization, outage prevention, reduction of maintenance work), the used ML approaches are mostly based on Artificial Neural Networks (e.g., Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), Feed-forward Neural Network (FNN)) and Support Vector Machines, which together make up 74% of the used ML methods in the Smart Grid domain [11]. These approaches all share a common need for large training data sets, which are provided by the big data infrastructure of the Smart Grid. The qualitative analysis [5] does present how this big data infrastructure is combined with Machine Learning. It is also already introducing applications of forecasting energy based on environmental influences like solar radiation and wind (also using state of the art ML methods).

In the end there are already lots of approaches which are incorporating data of external influences into the analysis and design of ML concepts, but nearly every publication [12–15] is focusing on a special use case and a small set of external influences. There are no existing concepts that attempt to examine the heterogeneity of a Smart Grid in its full glory.

2.2 Concepts for Pattern Recognition and Machine Learning

Machine Learning tries to detect structures and patterns in data. Its focus lies on the automatic detection of descriptions, regularities and connections in between the data. All this extracted knowledge can subsequently be used to solve the relevant problem. This step-by-step approach is called an inductive process, where the task is solved by a computer which tries to learn and understand the problem. In the literature and research field of machine learning [16–20] one will observe a diverse amount of algorithms and approaches, which are able to automatically learn to solve such tasks. Generally such algorithms are driven by data and in most of the cases the quality of the solution increases with the amount of available data. Typically, a machine learning model,

$$\hat{\mathbf{y}} = g(\mathbf{x}) \tag{2.1}$$

is composed of parametric models like artificial Neural Network architectures (e.g., linear model, perceptron, logistic regression, softmax regression) or non-parametric models which are based on probabilistic methods. The model $g(\mathbf{x})$, in the multivariate case, takes a feature example $\mathbf{x}$ as input and

applies these features to the ML structure with the corresponding model characteristics. As a result we observe the output $\hat{\mathbf{y}}$, which provides some information we are interested in.
The problems that can be solved with Machine Learning are classifying data, structuring data, compressing data, visualization of data, filtering data, selecting relevant information, extracting dependencies between data components, predicting new values and this may just be an incomplete list, but illustrates the broad range of Machine Learning applications.

2.2.1 Unsupervised Machine Learning

Unsupervised Machine Learning is used to generate structure from the data without any prior knowledge. So in this case we do not have any information about the targets we would like to observe. This methods simply try to extract structure of the data, to represent data in a simplified way, or to build models of the data generation process. Applications of unsupervised ML are **clustering** methods (e.g., k-means, mixture models, hierarchical clustering), **density estimation** (e.g., kernel density estimation, gaussian mixtures), **projection** methods (e.g., independent component analysis, principal component analysis, factor analysis) or **generative models** (e.g., belief networks, hidden markov models).
The following subsections will give a brief overview of the unsupervised ML concepts which are used in this work.

Unsupervised clustering

Unsupervised clustering is a commonly used exploratory analysis method, since it provides an idea about the structure of the data. Basically it can be defined as searching for subgroups with similar behavior. The decision about how to assign each data point to a subgroup follows the rule that all the data points within a cluster have to be as similar as possible to each other. According to some similarity measure one can assign each data point to its corresponding cluster.
How to choose the number of clusters and how to decide which similarity measure is used for assigning the data points to the clusters is application specific and also dependent on how the data is distributed. To be able to apply clustering one has to define the number of clusters K and the input data for the clustering algorithm. In most of the cases the input data is related to features of the raw data or samples of the raw data itself. In the following we will introduce the most used clustering algorithm K-means and Gaussian Mixture Model clustering.
K-means clustering is a distance based approach whereas a distance measure from each cluster centroid to each input feature is used to measure the similarity to this cluster. Basically the algorithm can be described in the following steps:

1. Choose the number of clusters K.
2. Initialize the centroids with K random data points.
3. Compute the distance measures between the centroids and all data points.
4. Each data point is assigned to the closest cluster.
5. Compute the new centroids by averaging over all assigned data points for one cluster
6. Iterate trough 3. - 5. until the centroids are stable and not changing any more.

Using this clustering approach one can find interesting structures in the underlying data, but as in every unsupervised approach we do not have a direct method for evaluating the model performance. Especially when it comes to choosing the number of clusters K. One method for determining the number of K is the elbow method. This method simply iterates through the K-means algorithm with an increasing number of K and calculates the sum of squared distances between the data points and their assigned centroids. Plotting the sum of the squared distances over the different values of K will result in an elbow-like shape. Normally one chooses K where this curve is at its bending point.
The goal of the **Gaussian Mixture Model** clustering is similar to K-means, both try to detect structure in the feature space. But taking a closer look at K-means one will recognize, that it does fit spherical boundaries between the different clusters in the feature space, whereas a Gaussian Mixture Model also takes into account the variance of the data. So basically one can describe the Gaussian Mixture Model as fitting K multivariate normal distributions to the data. The result is therefore a multivariate probability density function for each cluster, with their corresponding means and variances.
In contrast to K-means where a hard classification is performed based on the spherical cluster boundaries, Gaussian Mixture Models use a soft assignment of the data points to the respective clusters. Since we are aware of the distribution function for each cluster, we can calculate the probability of each data point to be drawn from each cluster. The data points are then assigned to the cluster where they have the highest probability of being observed.

2.2.2 Supervised Machine Learning

The opposite to unsupervised ML is supervised ML. In this case we do have information about the target outputs of our model and this so called labels are used for training. Typical applications are regression and classification. Regression tries to generate new values on a continuous scale, based on the past and present values. Classification is present in the discrete domain and we divide the resulting output space in a discrete number of $classes$ where one tries to assign each input feature to the corresponding class.

Linear Neural Network - polynomial regression

A very simple application of supervised Neural Network (NN) is the following linear model (s.c. neuron),

$$\hat{y} = g(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T\mathbf{x}. \tag{2.2}$$

If we were to write $\hat{y}$ and $\mathbf{x}$ as scalars ($\hat{y}$ and x), then we would have simply used the model for a least square line, but in our case, we speak of multivariate linear regression. This means that the model is able to learn a linear regression in the feature space. An interesting extension of this linear model is the consideration as polynomial regression. For this purpose we construct the parameter vector and the feature vector in the following way,

$$\mathbf{w} = \begin{bmatrix} w_0 & w_1 & w_2 & \dots & w_k \end{bmatrix}^T, \quad \mathbf{x} = \begin{bmatrix} 1 & x & x^2 & \dots & x^k \end{bmatrix}^T. \tag{2.3}$$

By choosing the order k of the polynomial function one can decide which $complexity$ of the data we want to reflect in our model. In terms of ML we also speak about the $capacity$ of the model, which is increasing with a higher number of parameters $\mathbf{w}$. The essential factor is to select the parameters of the model in such a way that the training data is reflected in an optimal form. For this, we have to measure how much the desired targets $\mathbf{y}$ differ from the actual results $\hat{\mathbf{y}}$. In the ML literature, this cost function is referred to the empirical error or risk function. For lots of Neural Network architectures this risk function is defined as the average of the loss function of a single training point. In this case we define the risk,

$$R_{emp}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(y^n - \hat{y}^n)^2 = \frac{1}{2}\sum_{n=1}^{N}(y^n - \mathbf{w}^T\mathbf{x}^n)^2 = \frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w}), \tag{2.4}$$

as the squared error loss between the target vector $\mathbf{y}$ and the predictions of the concatenated feature vectors $\mathbf{X}$. Normally for learning the optimal parameters, gradient decent is used for searching the minimum in the cost function, but in this linear case we are able to derive a closed form solution for the optimal parameter $\mathbf{w}^*$ in the form,

$$\mathbf{w}^* = \underset{w}{\operatorname{argmin}} \quad R_{emp}(\mathbf{w}) = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{2.5}$$

This Matrix $\mathbf{X}^+ = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, is also called the $pseudo\ inverse$ of the feature matrix $\mathbf{X}$.

2.2.3 Supervised Deep Learning

Basically there is no unique distinction between ML and Deep Learning (DL), since ML has experienced a transition to DL because of an enormous increase in data and computing power. Especially since 2010 [21], there was a huge improvement in the core competency of DL, which is the ability to learn abstract representations in the data. Reviewing DL literature [22–24] on will find an enormous amount of different architectures but the most used networks types are:

- Recurrent Neural Networks (RNNs): mostly used for sequential data (e.g., text, speech, videos)
- Convolutional Neural Networks (CNNs): image processing and recognition
- Feed-forward Neural Networks (FNNs): standard classification or regression problems

One of the most used data types in the context of Smart Grids is corresponding to time series data related to the grid measurements. Since Recurrent Neural Networks are commonly used for sequence analysis, it is of great interest to apply these kind of networks for analysing sequences in the grid measurements. The next section is therefore dedicated to introduce the main characteristics of a RNN which is called Long Short-Term Memory.

Long Short-Term Memory - LSTM

So far we encountered and described ML models and Neural Network architectures which are used for processing data which is supposed to be drawn from a distribution. Additionally we also assumed that all the data samples are i.i.d. (independently and identically distributed). Traditional Feed Forward Networks (e.g., the Linear Neural Network, Multi-Layer Perceptron) are very useful to learn patterns and structures from this data, but unfortunately this i.i.d assumption is not true for lots of data. Just considering the paragraph written here, one will agree that if the words would be randomly arranged, one will no longer be able to draw any meaningful conclusions of the resulting sentences.
This is one of the reasons why RNNs were developed. In principle they can be seen as Feed Forward Networks, where new activations (network layer outputs) depend on old activations from prior steps. This means that the current output depends on both, the current input and old activations. This gives this kind of networks the ability to learn dependencies and relations in sequences like text, audio, video or any kind of time series. While a Feed Forward Network maps the input $\mathbf{x}$ to an output $\hat{\mathbf{y}}$ with $\hat{\mathbf{y}} = g(\mathbf{x}, \mathbf{w})$, a RNN is mapping the input sequence $\big(\mathbf{x}(t)\big)_{t=1}^{t=T}$ to an ouptut sequence $\big(\hat{\mathbf{y}}(t)\big)_{t=1}^{t=T}$ by

$$\hat{\mathbf{y}}(t) = g(\mathbf{x}(0), \mathbf{x}(1), \ ..., \ \mathbf{x}(t); \mathbf{w}). \tag{2.6}$$

As basic RNN architectures e.g., Jordan Networks, Elman Network, Fully Recurrent Networks, do suffer from sequence learning problems like numerical instability, modern RNN architectures like the Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) Networks are used. This modern RNN variants do address the problem of storing the information over a long time period considering a short term input. The LSTM is therefore a Reccurrent Neural Network architecture with a memory cell for storing information. These memory cells are also controlled by gates which control the flow of information into the memory cell and out of the memory cell. Basically there exists an input gate which controls the flow of information into the memory cell and an output gate which learns to access the memory cell for outputting information. The LSTM cell (memory cell) basically consist of the following components,

- g: the memory cell input activation function for $\mathbf{z}(t)$,
- $\mathbf{i}(t)$: the input gate with sigmoid activation function,
- $\mathbf{o}(t)$: the output gate with sigmoid activation function,
- $\mathbf{f}(t)$: the forget gate with sigmoid activation function,
- $\mathbf{c}(t)$: the self-reccurrent unit,
- $h(\mathbf{c(t)})$: the memory cell activation function with the cell state **c(t)**,
- $\mathbf{y}(t)$: the memory cell activation,

A very popular and in most of the frameworks implemented version is the Vanilla LSTM which also introduces a forget gate in order to reduce the problem of indefinitely growing cell states. In vector notation the forward pass of the vanilla network is,

$$\begin{aligned}
\mathbf{i}(t) &= \sigma\big(\mathbf{W}_i^T + \mathbf{R}_i^T \mathbf{y}(t-1)\big) \\
\mathbf{o}(t) &= \sigma\big(\mathbf{W}_o^T + \mathbf{R}_o^T \mathbf{y}(t-1)\big) \\
\mathbf{f}(t) &= \sigma\big(\mathbf{W}_f^T + \mathbf{R}_f^T \mathbf{y}(t-1)\big) \\
\mathbf{z}(t) &= g\big(\mathbf{W}_z^T + \mathbf{R}_z^T \mathbf{y}(t-1)\big) \\
\mathbf{c}(t) &= f(t) \odot \mathbf{c}(t-1) + \mathbf{i}(t) \odot \mathbf{z}(t) \\
\mathbf{y}(t) &= \mathbf{o}(t) \odot h(\mathbf{c(t)}),
\end{aligned}$$

where $\mathbf{i}(t), \mathbf{o}(t), \mathbf{f}(t), \mathbf{z}(t), \mathbf{c}(t), \mathbf{y}(t) \in \mathbb{R}^I$. The vector $\mathbf{x}(t) \in \mathbb{R}^D$ corresponds to the external input and $\mathbf{y}(t-1)$ describes the memory cell activations of the last time step. The hidden layers are of dimension I and the inputs to this layer are of dimension D.

The vector $\mathbf{i}(t)$ is the input gate activation, $\mathbf{o}(t)$ is the output gate activation, $\mathbf{f}$(t) is the forget gate activation, $\mathbf{z}(t))$ is the cell input activation, $\mathbf{c}(t)$ is the cell state and the vector $\mathbf{y}(t)$ is the current memory

cell activation. The function g is the cell input activation function and the function h is the memory cell activation function. $h(\mathbf{c})$ is the memory cell state activation. The operator $\odot$ is the Hadamard's product, that is a point-wise (or element-wise) vector product.

While LSTM Networks are excellent for sequence analysis, they are also sometimes tricky to handle. First of all while designing an architecture one should be aware of the underlying problem and adjust the architecture according to the task. The choice of the adequate inputs to the network, the number of hidden layers, the gate activation functions, the cell state activation function can have a huge impact on weather the model is able to learn the task or not.

Also interesting to know is that when using Deep Learning frameworks to implement such architectures, we do not really need to take care about the learning algorithm itself, but it is worth mentioning that RNN architectures in general are trained in a similar way like traditional Neural Networks using backpropagation and gradient descent. But as we do have time dependencies one has to apply a trick which is called *unfolding in time* in order to also be able to backpropagate the error through the layers of a LSTM. Reviewing the literature one can define the following tricks which are very helpful when designing and using a LSTM architecture:

- **Forget gate** - It is very useful when one wants to focus on recent inputs, e.g., next time step prediction in a time series prediction task. For learning long term dependencies like predicting the winner of a game, one will not use a forget gate since it simply introduced difficulties for learning such long tasks.
- **Scaling of** g **and** h - Investigating long sequences where the elements which are relevant for the task are occurring just a few times, one can scale the memory cell activation g with αg ($\alpha > 1$). This causes this rare patterns to contribute to the memory cell. What is also beneficial is to use αh instead of h. This makes small changes in the memory cell recognizable.
- **Linear** g **and** h - A linear activation function can be helpful if one wants to perform a counting or accumulation task in the input sequence.
- **Sigmoid vs. tanh for** g - The sigmoid activation function is useful for detecting single patterns that are rarely appearing in the input sequence, while tanh is very useful if we want to learn for or against a fact, e.g., predicting an event based on the previous sequence elements.

Chapter 3

Methodology

"Data, perhaps the most important and at the same time the best guarded asset of our time."
Introducing this statement is a bit controversial, because to a certain extent it reflects the reality. If you look at the literature [2–5], in nearly every publication related to data analysis in Smart Grids, it is mentioned that the large amount of available and heterogeneous data must be used to extract knowledge and patterns. In reality, however, it is not always that easy to get access to this data. Especially when talking about heterogeneous data in Smart Grids which comes from various sources and different domains. All the diverse properties that are thereby obtained e.g., different sampling rates, units, standards, discrete and continuous measurements, have to be known and taken into account, in order to successfully implement a pipeline for generating knowledge.
This chapter will therefore propose a general methodology which can be used in order to tackle the problem of Pattern Recognition in such a highly dynamical and multivariate scenario like a Smart Grid. Basically it consist of two major parts. The first part will provide some approaches how to actually define and structure the heterogeneous Smart Grid environment in order to be able to extract information and patterns. Once the environment and the according data are defined one can execute the step of finding patterns with the help of data analysis methods. The second part is dedicated to actually use this generated knowledge for designing Machine Learning applications based on the extracted patterns. These Machine Learning concepts can have different aims and are highly depending on the underlying pattern, but basically one wants to either improve the pattern recognition task beyond the limits of human perception or design methods for improving the performance and efficiency of the Smart Grid.

3.1 Extracting patterns from a heterogeneous Smart Grid

Talking about data analysis and pattern recognition we will face one of the most important points right at the beginning. Especially in the case of a heterogeneous Smart Grid the first step towards a successful result is to define the environment with all its components, participants and possible environmental influences. The general overview of such a heterogeneous environment and all its possible influences is already discussed in section 1.1.

Once we managed to define the environment where we would like to extract patterns, we actually have to apply data analysis in all its variants. But applying data analysis can be a very challenging task, since the possibilities and available methods are enormous. To tackle this field we do have to follow a certain guideline. This can be formulated with the following steps:

1. **Asking the right question** - What is the goal of the data analysis? What objective should be defined in order to fulfill the given goal?
2. **Data gathering** - Which data is needed/available in order to be able to fulfill the goal? Where are we able to collect the data? How can we represent the data?
3. **Data Preprocessing** - Data cleaning, data accessing, combining/structuring data, are important steps to bring the data in an suitable form for analysis.
4. **Applied exploratory analysis** - trying to extract statistical properties, correlations, regularities and generating visualizations will extract events, patterns or information from the data. This knowledge can later be used to train ML algorithms for detecting these patterns.
5. **Conclusion** - Did the analysis answer the prior asked question? What are the conclusions of the data analysis?

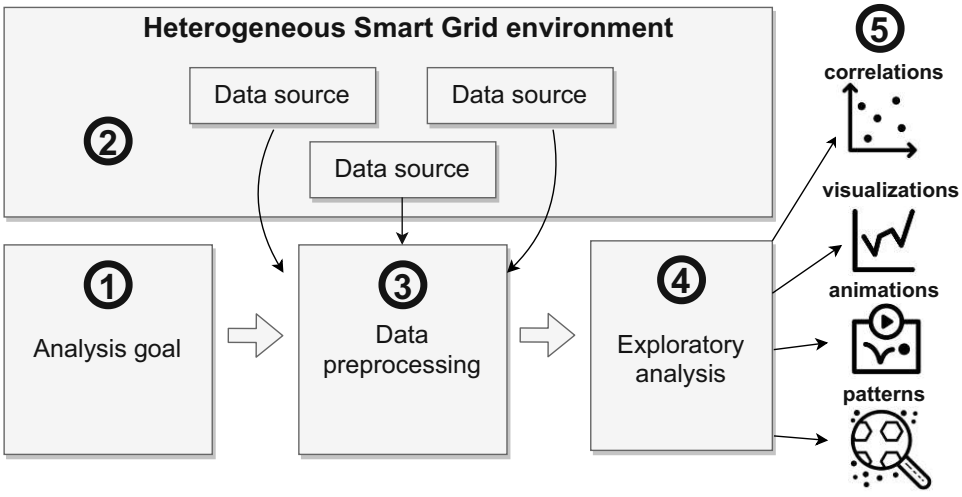

Figure 3.1: Proposed pattern extraction pipeline

Figure 3.1 illustrates the proposed steps for performing the task of extracting information and patterns from a heterogeneous Smart Grid environment and illustrates its modular approach. Each of these five blocks has its own purpose and is adaptable without the need of changing the structure and function of the remaining blocks. Especially when considering the block of gathering the data which is representing the heterogeneous environment it is of great importance to actually be able to add and remove different data sources without any obstacles.

Despite the fact that we know that the goal of finding patterns in a heterogeneous Smart Grid environment is quite broad, we still consider this as a starting point since in the first step where there is no existent information neither a definition of a specific problem, one has to apply different approaches in order to extract patterns, information and problem definitions. Once we are aware of problem definitions we can actually think of implementing ML concepts based on this extracted patterns.

In order to get a qualitative example of how to apply such a pattern extraction pipeline in the context of a real-world Smart Grid environment, we will provide a broad overview and answers to all the prior mentioned data analysis steps in chapter 4, which makes use of the historical data from the Smart Grid testbed Aspern[1] in Vienna, Austria.

3.2 Defining Machine Learning concepts based on patterns

The steps of the data analysis which can be highly influenced by the perception of the data analysts should be handled very carefully and also executed in an iterative manner. But once the person in charge of the analysis has managed to define the environment, gathered all the corresponding data and finally extracted some valuable information, we can think of developing Machine Learning and Pattern Recognition concepts which are related to the extracted information and knowledge of the exploratory data analysis.

One of the most important points before even starting with implementing Machine Learning concepts, is to formulate problem definitions, since the choice of the concept and method is highly dependent on the task to be fulfilled. These problem definitions can be related to patterns which are revealed during the data analysis steps. Patterns are most likely recognized by human perception via different visualizations, correlation or time series representations. It is therefore of great interest to actually formulate Pattern Recognition approaches, which can make use of this manually acquired knowledge to actually generate more information in an semi-automated way. So basically the idea is to use the cognitive capabilities of us humans for finding interesting patterns or at least hints for patterns. Once the information about the potential problem is present one can make use of the enormous different possibilities for implementing and training algorithms which are able to interpret the data or even improve the pat-

[1] https://www.ascr.at

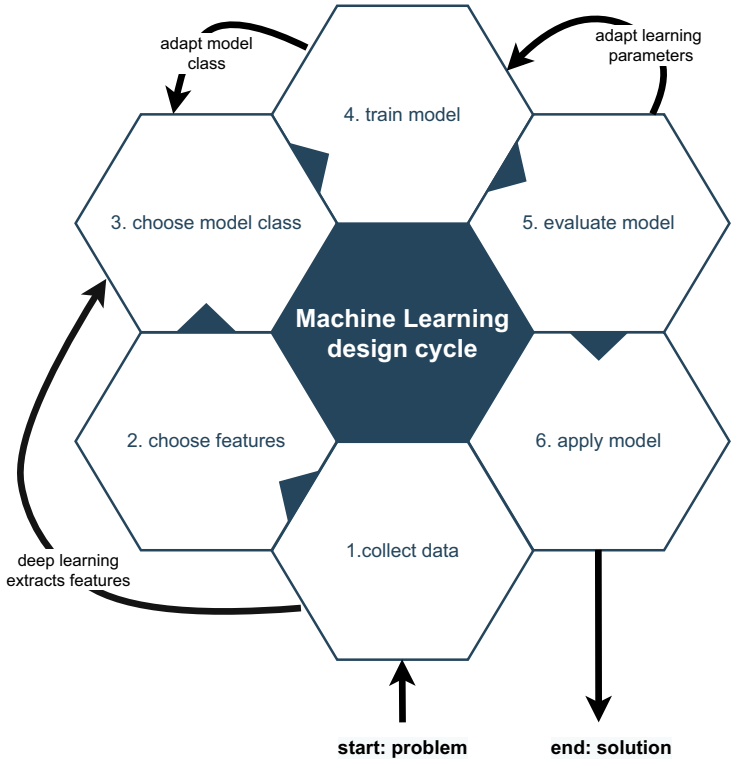

Figure 3.2: Proposed Pattern Recognition/Machine Learning design cycle

tern recognition and data analysis results. Reviewing the literature [17], one will recognize that once we decide ourself to implement a ML concept for solving a problem we will face an numerous amount of different variants and approaches. To overcome this issue we propose a guideline for designing ML and Pattern Recognition concepts. Figure 3.2 illustrates the proposed ML design cycle and all the steps which are necessary in order to develop such concepts. The general scheme of providing such a design cycle is motivated by [17] and the modified and proposed steps are listed in the following.

1. Collect data - Entering the ML design cycle with a problem, we do need enough information (data) which is representing the issue to be solved. Without data one is not able to apply any kind of ML. So the main goal in this first step is to define the problem and to collect the data which is containing any information regarding the problem. Nowadays, however, due to the increased occurrence of broadly collected data, it is also often the case that people are not aware of the specific problems and they simply want to detect irregularities and patterns. But in any case a sufficient amount of data has to be collected in order to be able to generate information.

2. Choose features - Once there is enough data collected, one has to decide which features or properties are useful to solve the problem. While $feature\ extraction$ denotes the process of creating features from raw data, a second possibility is that the raw data already consists of a large number of features and the relevant ones have to be selected. The better the $feature\ selection$ is adapted to the problem, the easier it is for the model to learn the task. At this point, a very important step is to use

as much a priori and domain knowledge as possible for selecting the features, as it can simplify the learning task.

While classical ML is dependent on this feature extraction and selection step, modern DL approaches do insert an additional functionality in order to automatically extract relevant features (e.g., filter learning in CNN's, dependency learning in RNN's). These approaches allow that less domain knowledge has to be invested, since the feature selection step can be skipped. However, it should be noted that this technique also comes with the disadvantage that in order to learn this feature extraction steps we need much more data compared to the traditional way.

3. Choose model class - Once the problem, the data and the corresponding features are defined we have to decide which model class we would like to use in order to solve the problem. Basically we distinguish between $parametric$ and non - $parametric$ model classes.

Parametric models represent the most used model class. In this class each parameter vector represents a unique model. Neural Networks and Support Vector Machines are typical examples, as the weights between the neurons or the support vector weights are representing parameters of the model. In most of the cases it is possible to effectively compute their derivatives. This gradient information, if pointing to a direction of improvement, is then used to move the weights through the parameter space towards an optimal solution (minimum of the gradient). The process of moving the weights through the parameter space is called learning.

Non - parametric models can be seen as the opposite to parametric models. This class takes the assumption, that the data distribution cannot be defined in a finite set of parameters and we usually think of this model class as a locally constant or superimposition of constant models. One could interpret the parameters of this class as functions. Examples of this non parametric models are, kernel density estimators and k-nearest-neighbors.

4. Train model - Once we have used our prior knowledge to choose a model class, an appropriate model has to be selected. This model selection process is based on the training set of the data and is therefore also often called $training$ or $learning$. When talking about learning we also need to measure the performance, as one has to decide if the network fulfills the desired requirements. ML literature therefore introduces two different performance measures, the empirical error and the generalization error, where the empirical error refers to the performance of the model on the training set and the generalization error describes the error of the model on future data. In the case of model selection, one would like to minimize the error on future data, but as can be assumed at this point, it is difficult to calculate this error, since future data is not available during the training. However, there are methods to approximate this performance measure, but in most cases the model which best explains or approximates the training set is selected.

An also very important topic when dealing with learning, is the fundamental differentiation between:

- $Unsupervised$ learning, where the desired outputs (targets) are not available during training and we can not immediately measure the performance of the model, and
- $Supervised$ learning, where the desired outputs (targets, labels) are available during training and used for updating the model.

5. Evaluate model - After training, or in many cases already during training, it should be evaluated how effective the selected model is at solving the underlying problem. A very important criterion is the choice of the so-called $hyperparameters$, which describe the complexity of the model and influence the model selection process. Typical hyperparameters are the model capacity (e.g., number of neurons, number of layers, number of weights) and the learning parameters (e.g., gradient decent, learning rate)

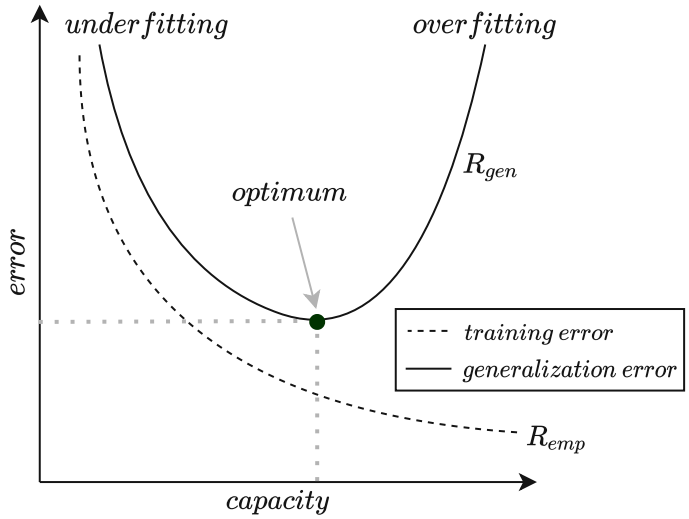

Figure 3.3: Overfitting, underfitting and capacity of a ML model

The trade-of between the error of the model and its capacity can be observed in Figure 3.3. On the left hand side we observe a model with a low complexity and the case where the training data cannot be fitted well enough. This case where achieving a low performance, regarding to a high training and generalization error, is called $underfitting$. On the other hand, when choosing a very complex model, we achieve an almost perfect explanation of the training data. If in this case the generalization error again results in a low performance, we describe this as $overfitting$. As already mentioned in the section before, we therefore have to find a trade-of between the model complexity and the performance in order to find a hyperparamter setting which is next to the minimum of the generalization error R_{gen}. This hyperparamter search is related to a separate research discipline, as it can be very complex to decide which parameters to use and how to tune them. Examples for this search techniques are trial and error,

grid search, random search, bayesian optimization and genetic algorithms.

6. Apply model - The selected model can now be used to solve the underlying problem. Regarding the real time capability it is not crucial if the model has been trained for a long time, but it is of great importance for the user how well the trained model can be integrated into his hardware framework, especially when the usability (e.g. waiting time) is taken into account.

Chapter 4

Exploratory data analysis in the testbed Aspern

To be able to actually apply the in section 3.1 proposed methodology and approaches for analysing a heterogeneous Smart Grid one has to have access to real historical data. The heterogeneous Smart Grid which is investigated in this thesis refers to the testbed Aspern[1] in Vienna, Austria. In general this testbed is divided into 12 measurement fields, whereas each field is dedicated to a distribution substation. In order to be able to successfully monitor and analyze the testbed, grid monitoring devices are installed in the individual measurement fields. These devices provide measurements of the grid values with a resolution of 2.5 min and are in this case available for a period of three years (2016 - 2019). The availability of these grid measurements is basically the starting point for implementing a successful data analysis and pattern recognition pipeline. The following sections are dedicated to the definition of the investigated heterogeneous environment, the implemented data analysis and the presentation of the extracted patterns.

Reviewing the proposed pattern extraction pipeline from Figure 3.1 we can see that we start with defining the goal of our analysis by asking the right question. This is a very crucial point and since our goal is to find patterns in a heterogeneous Smart Grid, we could argue that this definition of the analysis objective is quite broad. But in any case also again considering a very abstract and high level of entering the data analysis it is fine to actually start the analysis with the goal of finding structures, correlations and dependencies regarding the heterogeneous nature of Smart Grid and its environment.

[1] https://www.ascr.at

4.1 Data gathering - historical heterogeneous data

Data gathering is highly important, since this step is essential for fulfilling the analysis goal. In this work, the challenge is to gather a data set which represents the heterogeneous Smart Grid environment of the testbed Aspern. Especially we focus on defining the data set in such a way that one can use common sense knowledge, like weather and calendar information. This data is easy to acquire but has a huge impact on the Smart Grid. The following subsections give a qualitative description about the data set used in this work.

4.1.1 Historical grid data

The available grid measurements at each substation, which are currently stored in a timeseries database (InfluxDB[2]), are related to a 3-phase low voltage grid section and include the root mean square values of voltages ($U1 - U3$), currents ($I1 - I3$), active powers ($P1 - P3$) and reactive powers ($Q1 - Q3$). Figure 4.1 illustrates an example timeseries measurement of a distribution substation for 24 hours. The first information which can be derived from this figure is that the voltage in this case is fluctuating in a range of 232V - 238V. It is also apparent that the reactive power is mostly negative, which indicates that the transformer behaves in a capacitive way at this time window. One can also see that the voltage prop at 12:00 influences the reactive power consumption (or vice versa).

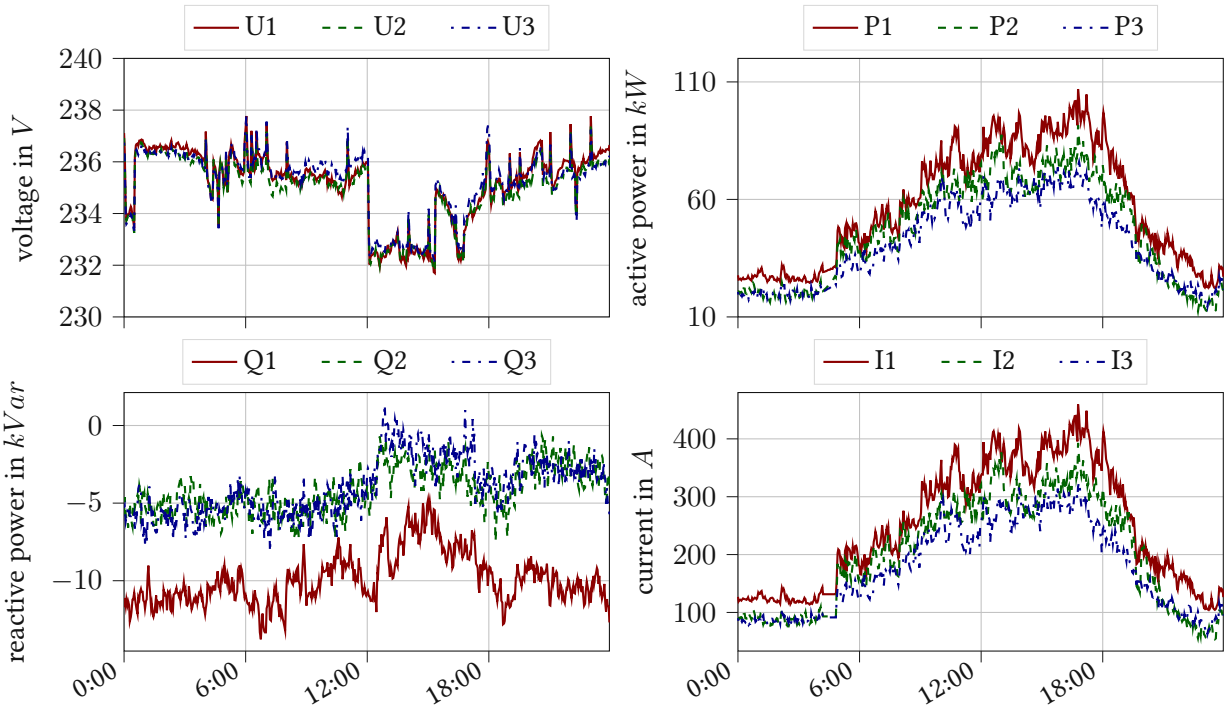

Figure 4.1: Example for a grid time series of one distribution substation

[2]https://www.influxdata.com

If we have a look at the active power over the duration of the day, we can clearly see a day profile. In the early hours from 06:00 onwards, the power increases steadily, whereas the inversion point is reached shortly before 18:00 and then the power decreases again with a negative slope. It is also worth mentioning that there is a significant difference in the rising and falling slope of the day profile. Comparing the active power and the current we might also recognize that the timeseries does have a similar curve shape, whereby this redundancy is related to the relationship $S = UI$.
By simply visualizing the time series data of a distribution substation for one day, one can see that there is already a huge amount of information which can be manually recognized by the observer. But since our main objective is to extract patterns and information in relation to the heterogeneous and environmental influences, we actually also do have to incorporate this data into the analysis. The next subsection will therefore describe the used environmental and heterogeneous data.

4.1.2 Corresponding environmental/heterogeneous data

As one of the goals in this work is to extract patterns from a heterogeneous Smart Grid, as described in section 1.1, the available heterogeneous data needs to be defined and described. As first step one could make a segmentation into external and internal data, whereas the external data in this work consist of:

- **Weather** (temperature, solar radiation, amount of daylight) - The temperature and solar radiation are observed from local senors in the testbed and are available with a sampling rate of 1 hour. The data about the daily amount of daylight is gathered with the help of the suntime[3] python library. The therefore used coordinates (latitude: $48.22573692335911°$, longitude: $16.508141942206418°$) are those of the Seestadt Aspern.
- **Calendar Information** - Data that is very easy to acquire but can have a big impact on the Smart Grid is the distinction between weekends, workdays and holidays. This information does have direct influence on the way people are producing and consuming energy. In this case the data is obtained with the holidays[4] python library.

One essential point for extracting and interpreting patterns is the knowledge of the internal structure of the grid. This includes the components, energy consumers, energy producers and the geographic distribution of the individual parts. This so called domain knowledge for the investigated testbed is listed below:

- **Battery storage systems** - Prevention of load peaks or storing the over-production of energy
- **Photo-voltaic systems** - Renewable energy source

[3]https://pypi.org/project/suntime
[4]https://pypi.org/project/holidays

- **Heating systems** - Temperature controlled heat pumps
- **Energy consumers** - Industry and commercial buildings, residential areas, schools, car parks
- **Geographic information** - Latitude and longitude coordinates of the distribution substations
- **Grid plan** - Description of the grid measurement concept with the information on how the individual substations are interconnected and distributed through the grid.

In the end one might ask the question why we are exactly using this proposed heterogeneous data. This question is indeed justified, since we are simply just able to detect patterns and extract information where we are also having access to the corresponding and underlying data. But having access to data is already the key point. Data which is measured with the help of sensors (e.g., emissions, traffic) might be actually hard to acquire in a spatially high resolution.

Whereas the weather (e.g., temperature, solar radiation) and calendar information are data sources where we do not need this high spatial distribution. This data sources are steady over a greater area and often even available via open source platforms.

Therefore the used heterogeneous influences are selected according to the motto *easy to acquire but huge impact.* In the end one has to make a trade-of between which data to include an which not, but anyway the modular approach of the proposed preprocessing and data analysis pipeline does allow the user to add additional data sources without the need of restructuring the pipeline.

4.2 Data preprocessing - combining and structuring the data

The third step in the data analysis pipeline is dedicated to the data preprocessing. Table 4.1 summarizes the different elements in the available data set and in order to be able to start with the actual data analysis, the individual parts should be combined to form a structured data set. The general flow of the proposed preprocessing and analysis pipeline is illustrated in Figure 4.2 this is in general an applied form of the pattern extraction pipeline in Figure 3.1. It consists of the following steps:

1. Choose the time window (start date, end date) and the distribution substation for the analysis.
2. The preprocessing script accesses the different data sources and combines the gatherd data to a uniform dataset, where the weather data is sub-sampled to fit to the sample rate of the grid data. Additionally the mean values of the three phase components of the grid data are also added to the data frame, e.g. $P_{mean} = (P1 + P2 + P3)/3$
3. For generating and visualizing trend information the previously extracted high resolution data is sub sampled to a daily mean data frame.
4. Normalized data frames are also added, in order to be able to compare different substations. The normalization in this case is based on zero mean and unit variance.

data	type	sample rate	components	unit	source
grid data - 12 distribution substations	time series	2.5 min	active power $P1 - P3$ reactive power $Q1 - Q3$ voltage $U1 - U3$ current $I1 - I3$	kW Var V A	influxDB
weather	time series	1 min 1 min	temperature solar radiation	$°C$ W/m^2	csv - file csv - file
calendar info.	categorical	–	"weekend", "working day", "holiday"	–	database

Table 4.1: Historical heterogeneous Smart Grid dataset

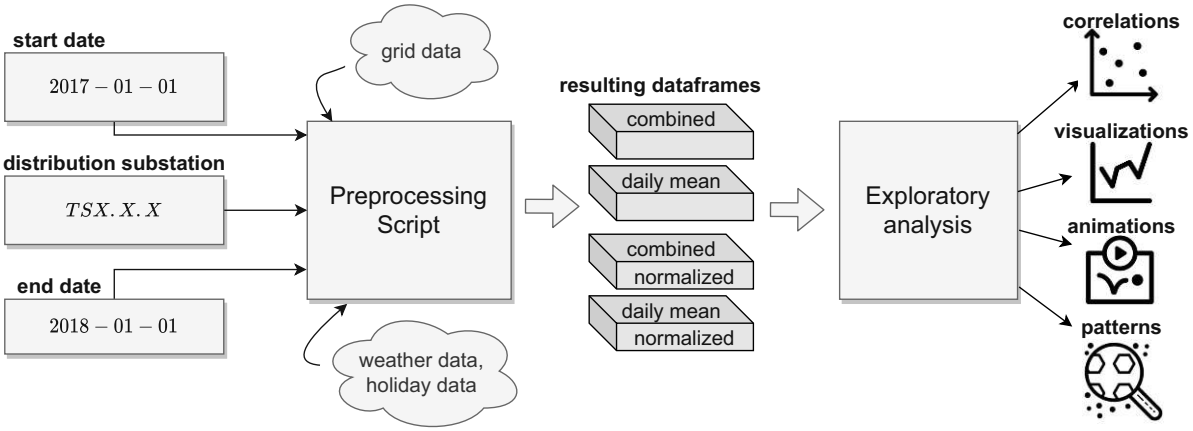

Figure 4.2: Proposed reprocessing and exploratory analysis pipeline for the data analysis task

4.3 Extracting patterns, correlations and visualizations

Once the data is preprocessed and available in a structured data frame, we can perform the actual exploratory analysis, this is referring to the right part of Figure 4.2. As we would like to compare the behavior and extract patterns from all the different distribution substations an automated information generation process is developed. This process extracts various visualizations and correlations for each substation. In this special case the analysis is performed for a 2 years time window (start date: 2017-01-01, end date: 2018-12-31), and the extracted visualizations/tables for each distribution substation including a dedicated example are listed below:

- Scatter plot of the daily mean grid measurements (Figure 4.3)
- Scatter plot of the daily mean measurements for temperature, solar radiation and the main grid measurements (Figure 4.4)
- Time series plot of the heterogeneous daily mean measurements (Figure 4.5)
- Correlation matrix for the heterogeneous measurements (Table 4.2)

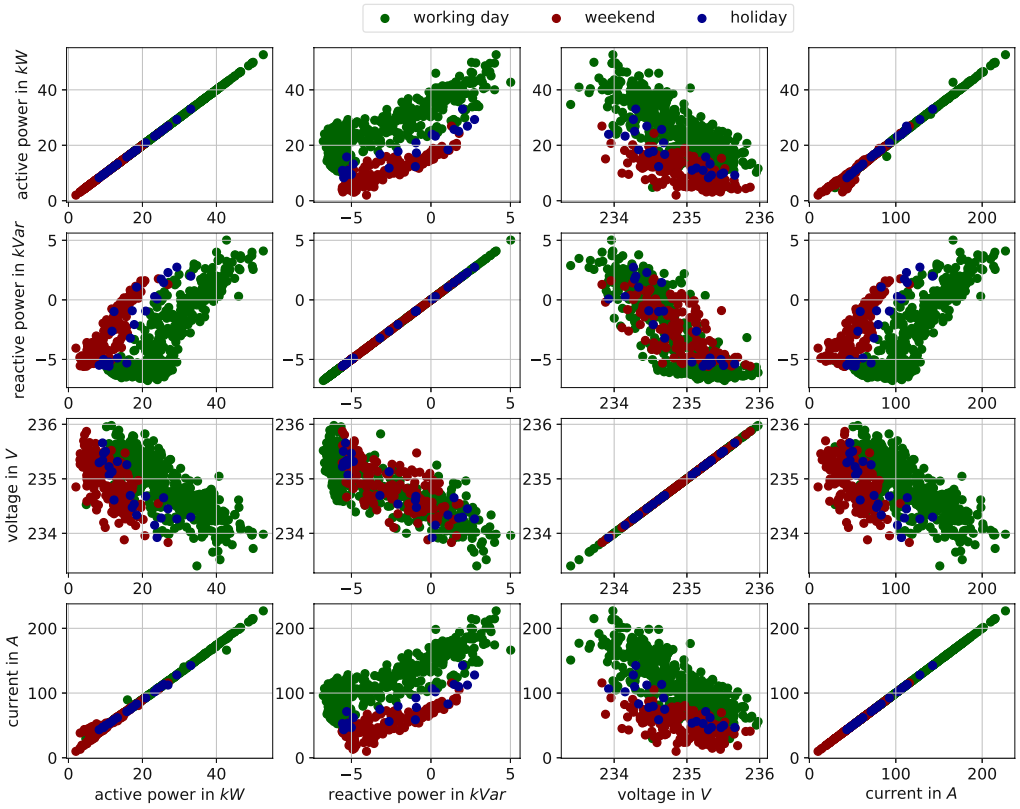

Figure 4.3: Example for a scatter plot of the daily mean grid measurements

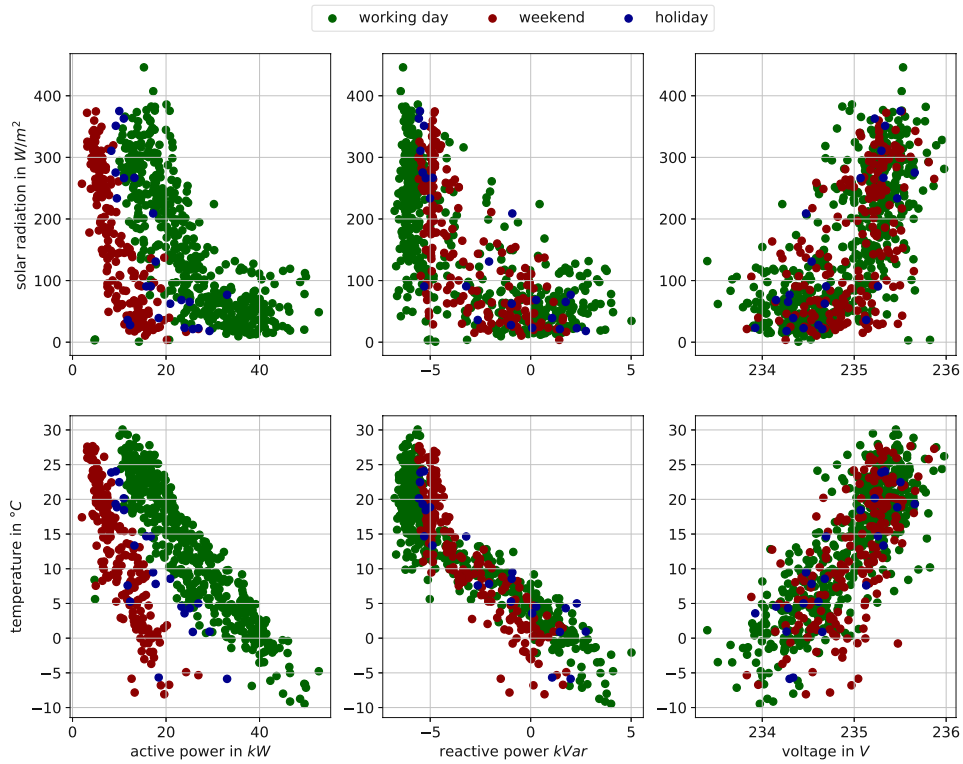

Figure 4.4: Example for a scatter plot of the power grid measurements

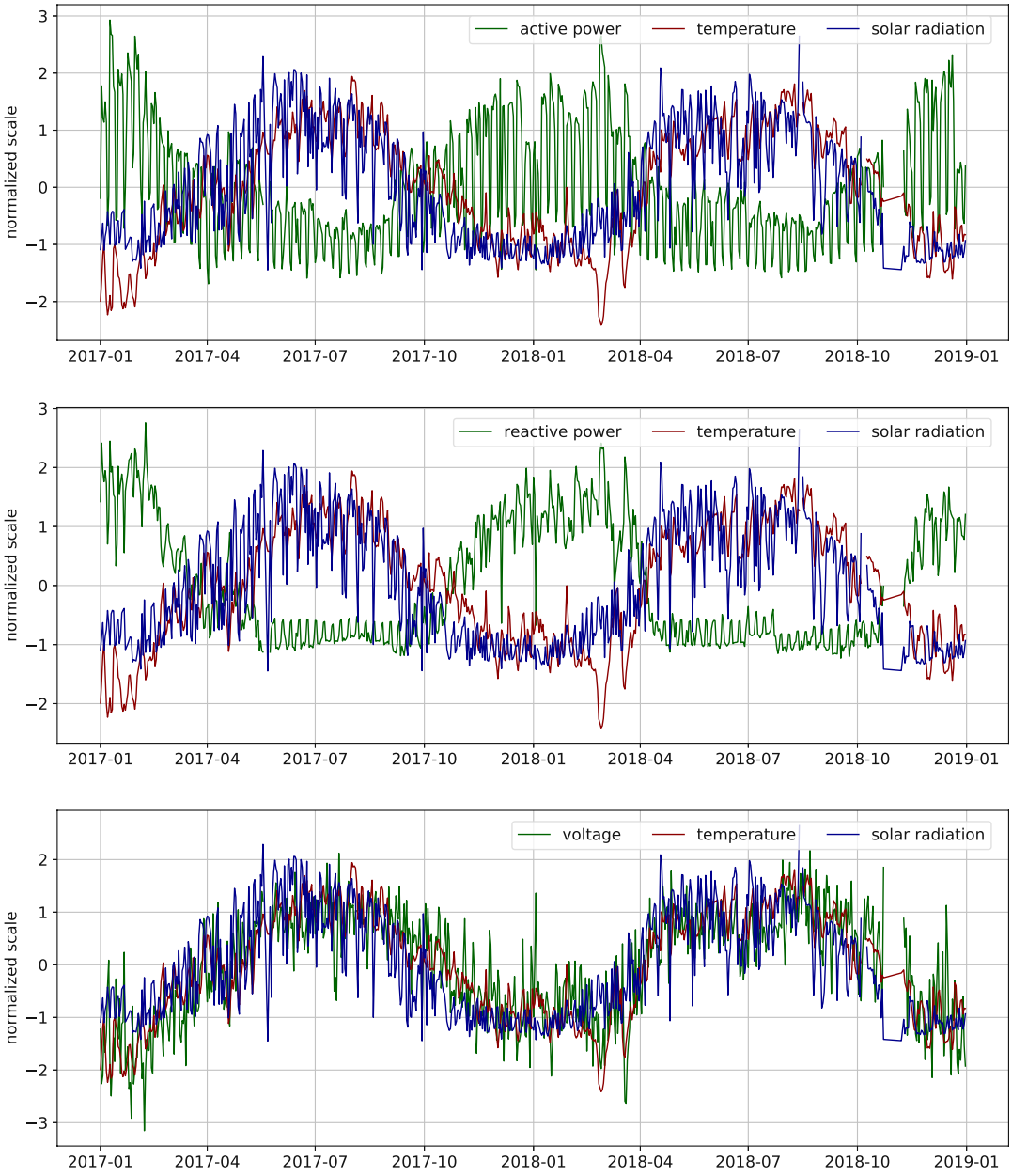

Figure 4.5: Example for a time series plot of the heterogeneous daily mean measurements

	I_mean	P_mean	U_mean	Q_mean	Temperature	Solar Radiation	Suntime
I_mean	1.000	0.997	-0.658	0.692	-0.668	-0.582	-0.607
P_mean	0.997	1.000	-0.662	0.693	-0.675	-0.591	-0.613
U_mean	-0.658	-0.662	1.000	-0.833	0.787	0.656	0.693
Q_mean	0.692	0.693	-0.833	1.000	-0.902	-0.712	-0.791
Temperature	**-0.668**	**-0.675**	**0.787**	**-0.902**	1.000	0.788	0.831
Solar Radiation	-0.582	-0.591	0.656	-0.712	0.788	1.000	0.839
Suntime	-0.607	-0.613	0.693	-0.791	0.831	0.839	1.000

Table 4.2: Example for a correlation matrix of all heterogeneous daily mean measurements

4.3.1 Animated time series representation - minutes over days

In order to visualise how the data and especially the time series are changing over the different days a day profile animation using the python library plotly [5] is implemented. Figure 4.6 shows the active power of all three phases plotted over the minutes since midnight.

By pressing the play button, one can monitor the animation of all the lined-up days (30 days in this case), one after the other. This method turned out to be highly useful to quickly look over the timeseries data and it also provides a useful way to visually identify irregularities or anomalies.

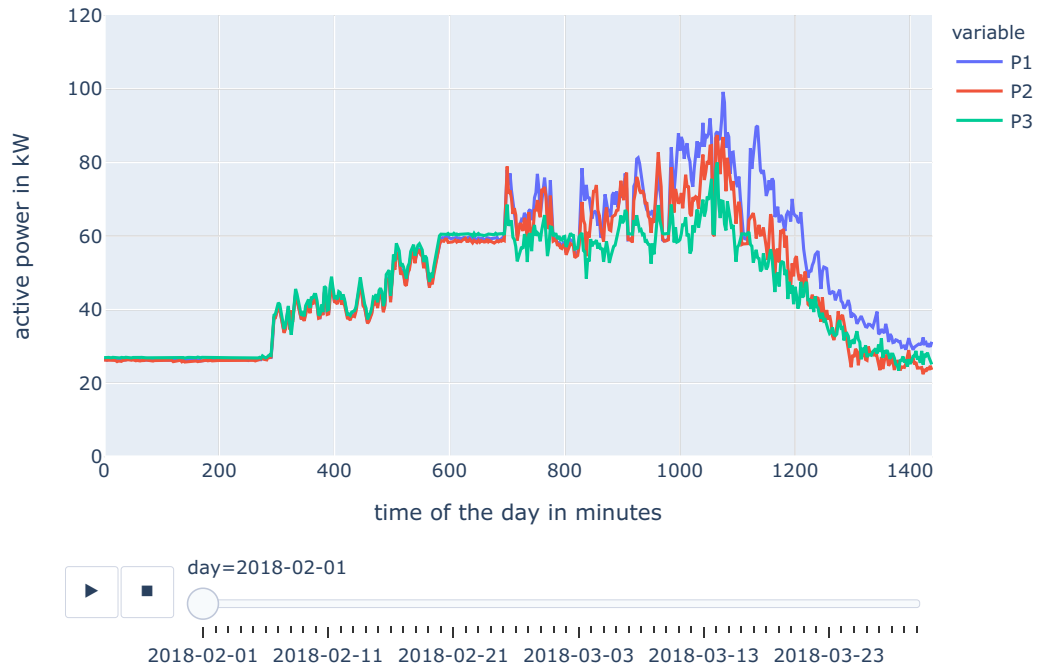

Figure 4.6: Day profile animation displaying the change in active power of all three phases

4.3.2 Spatial representation

By investigating the grid and the different time series for each substation it is also very interesting to get a feeling and an overview how the energy consumption is distributed over the grid. Especially, the spatial distribution is of great interest. To achieve an insight of the temporal changes in the spatial energy consumption, a tool for creating animated visualizations of the spatial energy consumption is developed. In this case the spatial information is chosen accordingly to the position in the grid plan (this could easily be replaced by geographic coordinates). Figure 4.7 illustrates an example where we can observe different bubbles which correspond to the different substations. The size of each bubble is

[5] https://plotly.com/

equal to their daily mean active power consumption. Again by pressing the play button one can observe the animation of the different lined up days (30 days in this case). Due to data protection reasons the original grid plan is replaced with a random map in this illustration.

Figure 4.7: Example for animated spatial consumption distribution (no real geographic locations)

4.3.3 Patterns in the daily mean time series

After automatically creating the analysis for each of the 12 substations, the visualizations are investigated and useful patterns are extracted. These patterns can later be used to optimise the grid and they are also very useful to define ML problems. The next sections are dedicated to the found patterns in the daily mean measurements.

Pattern 1: Power consumption difference between working days and weekends

Investigating Figure 4.3 and focusing on the scatter plot between the active power and the reactive power, one can clearly recognize, that there is an unique distinction between the power consumption during the week (marked in green) and during the weekend (marked in red). Additionally one can see the power consumption on holidays (marked in blue) which is also referring to lower values compared to during the week.

This information is quite interesting and its root cause seems familiar. When checking the domain

knowledge for this substation it turns out, that the main consumers are a school and a kindergarten. So there are simply less consumers present on the weekends. But anyway, this knowledge about the switching behaviour in the power consumption can be very useful when trying to optimize modern energy communities.

Pattern 2: Temperature dependent active power consumption

The next pattern is extracted from Figure 4.4 and focuses on the relation between the active power and the temperature. In addition to the distinction between weekdays and weekends one can clearly see a negative correlation between the temperature and the power consumption. In this special case the correlation coefficient equals $r = -0.675$. This inverse behaviour can also be recognized by focusing on the time series representation of the active power and the temperature (see Figure 4.5) .
This pattern can be caused by heating systems (e.g., heat pumps) and the correlation is especially high for substations where there is no battery system installed. The installed battery systems, cover up this pattern, as they are supposed to smooth out such fluctuations and especially grid peaks.
The knowledge about the temperature dependent power consumption can be used to make predictions based on temperature forecasts, or if this temperature dependent behaviour is not welcome, this information about the correlation can be used to install and design battery systems that compensate this behaviour in an efficient way. Especially one could design the charging and peak shaving strategy in a temperature dependent manner.

Pattern 3: Temperature dependent reactive power consumption behaviour

After analysing all the visualizations and correlations of all substations, four different reactive power consumption patterns are identified:

- Temperature dependent reactive power consumption clusters
- Combination between a negative and positive correlation between the temperature and the reactive power
- Positive correlation between the temperature and the reactive power
- No correlation between the reactive power and the temperature.

While the first two patterns can be related to temperature depending heating pump activities, the third one could be caused by a cooling system.

Pattern 4: Solar radiation dependent active power consumption

When comparing the correlation coefficients of substations where a photo-voltaic system is installed, there is always a stronger correlation between the solar radiation and the active power. Investigating Figure 4.8 one will observe exactly this behaviour. We can observe how the daily mean values of the solar radiation and the temperature are influencing the active power consumption. On the left in red we can observe the influence of solar radiation with a correlation coefficient of $r = -0.88$. The influence of temperature on the active power consumption (in green) does result in a lower correlation of $r = -0.72$. It is also very interesting to spend a look at the right subplot where we can observe the time series and the seasonal behaviour of the active power and the solar radiation. It again does reflect its inverse behaviour.

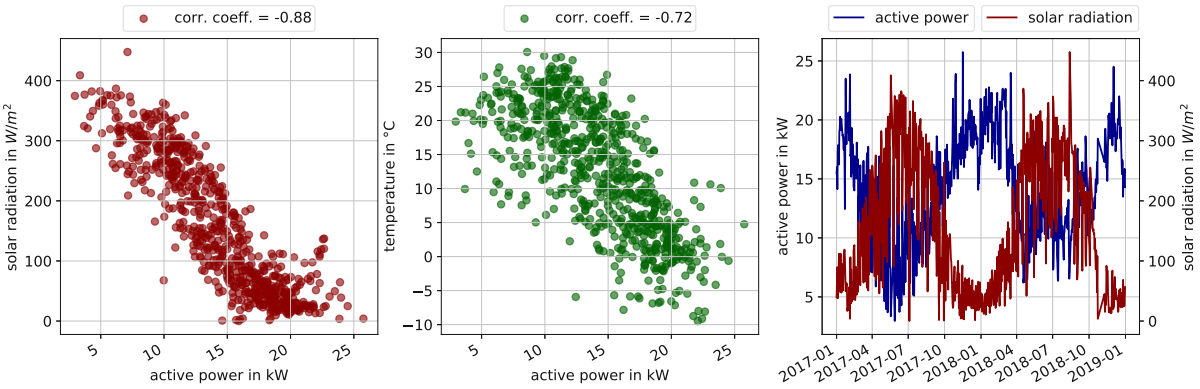

Figure 4.8: Example of solar radiation and temperature influence on the active power

Anomaly 1: Outlier in the feature space

Taking into account the feature space $X = [I_{mean}, P_{mean}, U_{mean}, Temp., Solar\ Radiation]$ and observing the extracted visualizations of all the different substations one will recognize outliers. These outliers can be related to anomalous behaviour of the grid. In order to determine their root causes (e.g., system failures or missing values in the time series) it is of great interest to investigate and detect their occurrence. Figure 4.9 represents such an example, where on the left we can observe a scatter plot of the active power and the voltage. The outliers in this case are recognizable around the elliptic point cloud. Taking a look at the second subplot we will observe a similar behaviour by observing the reactive power over the current. To determine the root cause of these outliers, it is also of great benefit to investigate the corresponding time series behaviour (illustrated in the right subplot). There we can already observe peaks in the active power consumption. A closer look at these peaks shows that on these days in this particular case there is an absence of measured values, which finally led to the outliers by calculating the mean values of the respective time series for these days. This knowledge is a perfect starting point to apply the ML design cycle (see section 3.2) in order to develop a concept which is able to detect these events.

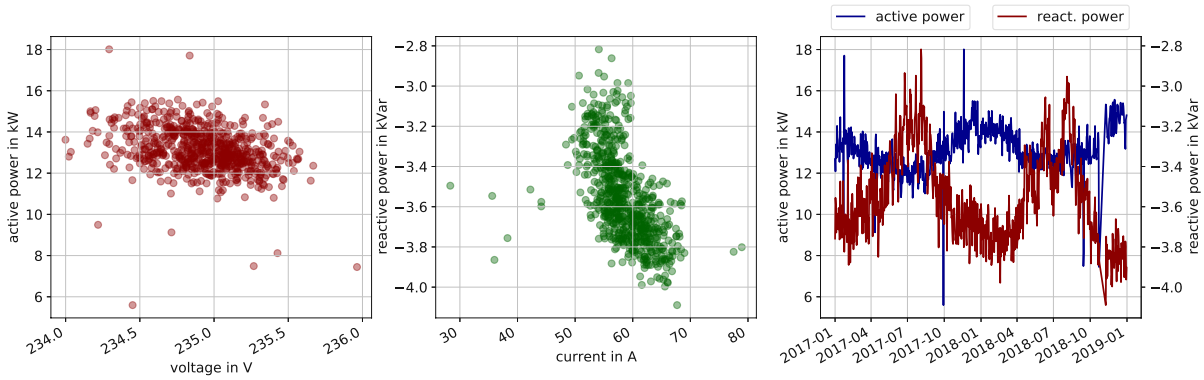

Figure 4.9: Examples for outliers in the daily mean values

4.3.4 Patterns in the high resolution time series

The patterns and anomalies extracted so far have a strong focus on analyzing the behavior of the grid based on the daily averages. This provides a very good insight which general trends are occurring during the year and it also provides useful information how the system is influenced by external factors. The aim of the the next section is to focus on the time series with high temporal resolution, since there might also be information or behaviours which were simply not visible in the daily mean analysis.

Pattern 5: Different day profiles because of the environmental influences

As already mentioned in section 4.1.1 and visualized in Figure 4.1, one can observe characteristic day profile curves in the active power consumption during the day. In addition to the classic behavior, where the consumption increases steadily in the early hours and decreases again in the evening after the change point, there are also different effects on the day profiles due to the external or internal domain-specific influences on the grid. Investigating the substations and their heterogeneous environment we do have the following influences on the day profiles:

- Calendar influence (working day, weekend)
- Heating system influence
- Battery storage influence
- Photo-voltaic influence

Figure 4.10 illustrates day profiles where one can observe the different environmental influences. On the left illustration one can see the difference in the day profiles between weekends and working days. This plot also reveals the information about steps in the power consumption which can be related to heating pumps. In general these steps will occur in correlation with the temperature or the corresponding temperature control system.

The second figure is related to the influence of the battery storage system. In red we can observe a system state where the battery system is limiting the power consumption to 60kW, whereas in the afternoon the storage seems to be completely discharged and the original power consumption profile is visible again. On the opposite, we can observe in green a profile where the battery is not influencing the power consumption at any point.

The third illustration in this case is related to a substation with an installed photo-voltaic system. It is obvious to see, that the consumption behavior is highly influenced by the solar radiation and during the day there is even a phase of energy overproduction.

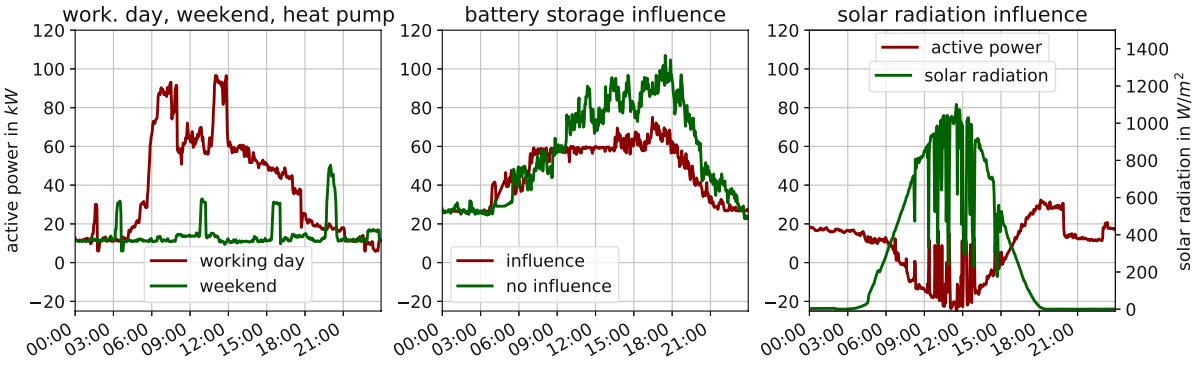

Figure 4.10: Examples for different day profiles because of the environmental influences

Anomaly 2: Irregular behaviour in the grid time series

The presented visualization methods for animating the different time series do also reveal interesting anomalies regarding the time series of the grid. One Example of such an anomaly is related to the active power time series consumption at a substation with an installed battery storage system. At this substation there exists a battery maintenance event, where the battery is fully discharged and charged during the day, this event occurs at a few random days in the year. One example of this anomaly is illustrated in Figure 4.11.

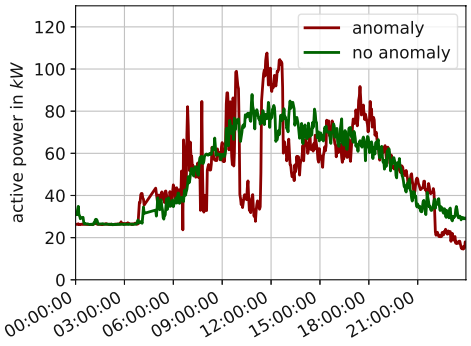

Figure 4.11: Example for a time series anomaly in the active power

4.4 Conclusion

To determine if we have solved the data analysis task and if we have properly answered all the prior asked questions, is a decision which is strongly depending on the viewpoint of the observer. By mentioning this sentence we have also introduced one of the the major problems of data analysis. The perception and the amount of prior assumptions, which are made before even starting with the data analysis, do have a strong influence on the results. Its therefore highly recommended to start at a unbiased and abstract level.

The starting point in this work is the availability of historical time series grid measurements from all the available distribution substations in the testbed Aspern. Additionally there is the availability of the grid plan which involves information about the different substations and their underlying components. This abstract starting point can now be tackled with the in Figure 3.1 proposed pattern extraction pipeline. The first step of defining the goal, as extracting patterns from the heterogeneous Smart Grid environment is followed by defining the environment and the available data. In this case the focus is strongly put on data which does have a strong influence and is easy to acquire (e.g., weather behavior and the calendar information).

Once the data sources are determined there is a need for structuring and combining the data. Figure 4.2 is therefore proposing a way of structuring, combining, analysing and visualizing the prior gathered heterogeneous information. This pipeline does extract analysis results for each substation in the investigated Smart Grid environment and reveals information, which is then used to extract and define patterns. These information and patterns are related to e.g., power consumption difference between working days and weekends, temperature dependent power consumption and different day profiles because of the environmental influences.

All these patterns do reveal the fact, that the Smart Grid is strongly influenced in its behaviour because of the environmental influence (e.g., temperature, solar radiation, calendar information) and its inherent components (e.g., photo-voltaic, battery storage, heat pumps). The plain knowledge about the existence of these patterns is very valuable, as it is the foundation for exploring the individual patterns in more detail. Additionally one can develop Machine Learning concepts which allow for detecting these patterns (e.g., anomaly detection, battery maintenance event detection), making predictions based on the heterogeneous information (e.g., power consumption prediction based on temperature and calendar information) or optimizing control strategies (e.g., calendar and temperature based heat pump control).

To summarize, we can state that the main benefit of the proposed method is the abstract starting point, which allows to apply this concepts to any new Smart Grid environment in order to extract hidden information which should be used to optimize and secure the grid.

Chapter 5

Pattern Recognition and Machine Learning concepts

The previous chapter is dedicated to analyse the heterogeneous Smart Grid environment based on historical data. This intensive data analysis step does reveal different information, patterns and characteristics. The first step of acquiring these knowledge is mostly achieved by manual observation. The second step in line, which should also include more automatism, is to use this knowledge for designing Machine Learning concepts e.g., power consumption prediction based on the heterogeneous influences, identifying and clustering of different system behaviours or detecting events in real-time. The following sections will introduce the proposed Machine Learning concepts and also explain each method with the results of a specified use case scenario.

5.1 Consumption prediction based on heterogeneous data

Considering distribution substations where the heterogeneous data like the temperature, solar radiation and the calendar information is influencing the behaviour and the consumption of the grid, one might be interested to actually use this information in order to create a function which allows to predict the consumption based on the heterogeneous data. This ML concept is therefore related to:

- Pattern 1: power consumption difference between working days and weekends
- Pattern 2: temperature dependent active power consumption
- Pattern 4: solar radiation dependent active power consumption

Reviewing the ML design cycle from Figure 3.2 we can define the underlying problem in this case as approximating a function

$$\bar{P}(\bar{T}) = g(\bar{T}, \mathbf{w}), \tag{5.1}$$

which allows us to calculate the daily mean consumption $\bar{P}$, based on the external measurement $\bar{T}$ and the model coefficients $\mathbf{w}$. The data collection for this issue is already achieved in chapter 4. The next step is to choose an appropriate model class for solving the task. In this case, as this problem is related to a two dimensional space $(\bar{P}, \bar{T})$ a linear neuron, which can also be described in form of a polynomial regression, is an appropriate choice. As the theoretical concept of this model class is already described in section 2.2.2, we can directly reformulate the coefficients and features,

$$\mathbf{w} = \begin{bmatrix} w_0 & w_1 & w_2 & ... & w_k \end{bmatrix}^T, \quad \mathbf{x} = \begin{bmatrix} 1 & \bar{T} & \bar{T}^2 & ... & \bar{T}^k \end{bmatrix}^T. \tag{5.2}$$

The feature matrix $\mathbf{X}$ is constructed as,

$$\mathbf{X} = \begin{bmatrix} 1 & \bar{T}_1 & \bar{T}_1^2 & ... & \bar{T}_1^k \\ 1 & \bar{T}_2 & \bar{T}_2^2 & ... & \bar{T}_2^k \\ . & . & . & ... & . \\ 1 & \bar{T}_n & \bar{T}_n^2 & ... & \bar{T}_n^k \end{bmatrix} \quad \in \quad \mathbb{R}^{n \times (k+1)}, \tag{5.3}$$

and the target values $\mathbf{y}$ as,

$$\mathbf{y} = \begin{bmatrix} \bar{P}_1 \\ \bar{P}_2 \\ . \\ \bar{P}_n \end{bmatrix} \quad \in \quad \mathbb{R}^{n \times 1}. \tag{5.4}$$

The variable n denotes the number of investigated days and k the order of the polynomial. Using this concatenated form of the daily mean consumptions and the polynomial expansions of the daily mean heterogenous measurements we can fit and select the model by obtaining the optimal parameter in the following way,

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{5.5}$$

5.1.1 Results - active power prediction based on temperature: substation 1

The result for the first example substation is visualized in Figure 5.1. The investigated time period for this analysis are two years: 2017-01-01 – 2019-01-01. The investigated substation shows a negative correlation between active power and temperature in lower regions of the temperature and a positive correlation if the temperature reaches higher values. As already visible by just observing the data points, there seems to be an overall quadratic relationship.

Once we have implemented the algorithm for fitting the polynomial function to the corresponding data, one can simply choose different orders of k and visualize the results.

In this case the order of the polynomial is chosen as k = 2 and we therefore acquire the following function for the daily mean power consumption prediction,

$$\bar{P}(\bar{T}) = g(\bar{T}, \mathbf{w}) = 45.35 - 0.35\,\bar{T} + 0.02\,\bar{T}^2. \tag{5.6}$$

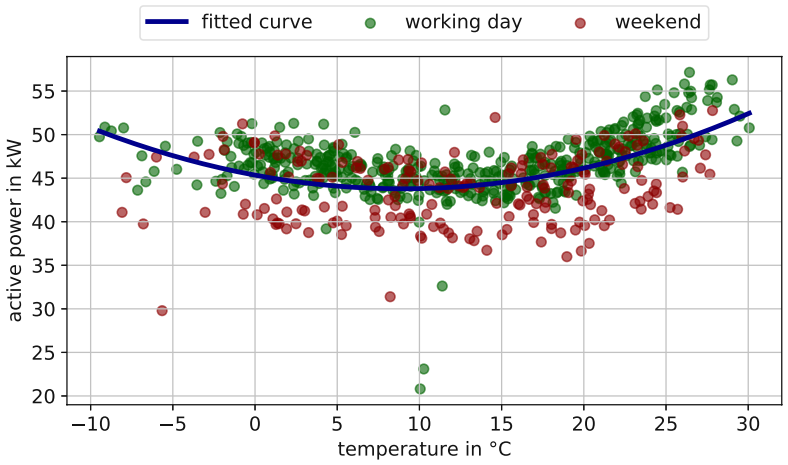

Figure 5.1: Power consumption prediction based on temperature for substation 1

Temperature is a quantity which is used in many applications and also in many cases available in the form of predictions. These temperature forecasts can now be used to predict the corresponding power consumption. The knowledge about the future power consumption can be used to optimize various control algorithms. Load flow controllers, which try to generate an optimal behavior between energy storage systems, renewable energy sources and the energy consumption of energy communities, are an ideal basis to include such predictions in the optimization of the control algorithms. One of the most important points is to make the prediction as accurate as possible. If we look at Figure 5.1 we see that the difference between weekdays and weekends is not crucial in this case, but as we have already seen in section 4.3.3 there are also substations where we find such a distinctive behavior. The next subsection will therefore illustrate how the incorporation of categorical calendar information can improve the prediction results.

5.1.2 Results - active power prediction based on temperature: substation 2

Again investigating the time period of two years: 2017-01-01 – 2019-01-01 and taking into account a substation where we do have a strong calendar related power consumption behavior we will observe that in order to generate an accurate prediction result we have to include the categorical information about the weekend and working days into the prediction task. This results therefore in fitting two

functions, as illustrated in Figure 5.2. One polynomial is representing the behavior on the weekends and the other one represents the behaviour during the working days,

$$\bar{P}_{work}(\bar{T}) = g_1(\bar{T}, \mathbf{w}) = 38.22 - 1.30\,\bar{T} + 0.013\,\bar{T}^2, \tag{5.7}$$

$$\bar{P}_{weekend}(\bar{T}) = g_2(\bar{T}\mathbf{w}) = 15.78 - 0.53\,\bar{T} + 0.003\,\bar{T}^2. \tag{5.8}$$

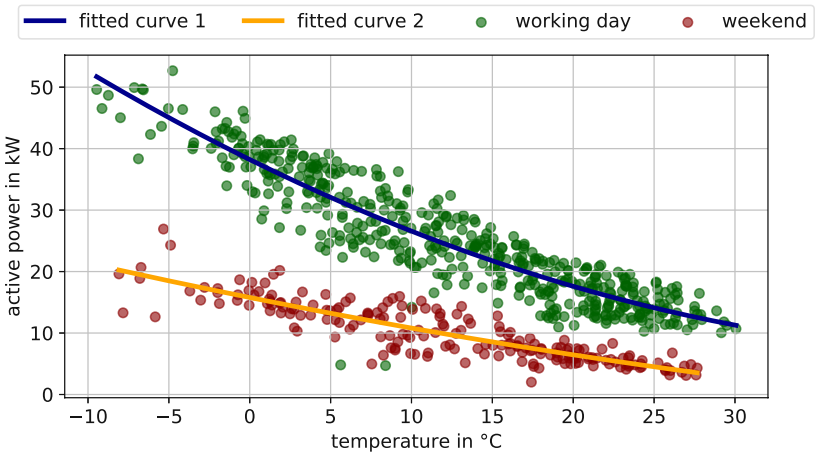

Figure 5.2: Power consumption prediction based on temperature and calendar info for substation 2

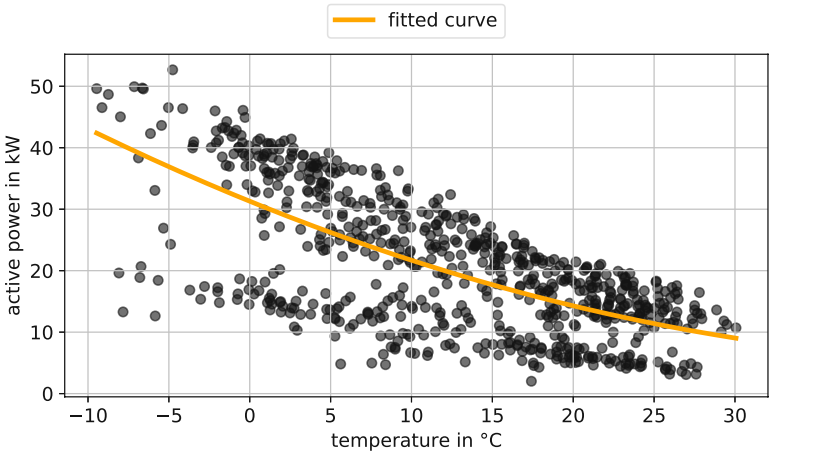

Figure 5.3: Power consumption prediction based on temperature for substation 2

Investigating Figure 5.3 where we are removing the calendar information from the prediction results on purpose does reveal the fact that without including all the available heterogeneous information we are producing weak prediction results. This indicates that if possible we have to incorporate as much heterogeneous information as available. Applications where one can incorporate this calendar and temperature related power consumption predictions are e.g., control algorithms for heat pumps, battery storage systems or dynamic price optimizations in energy communities.

Improvements and challenges of this method

As in every ML fitting or training process, one has to make assumptions about the model complexity. In this case when we talk about the complexity we have to determine the order k of the polynomial function and also the number of functions which should be fitted according to the categorical dependencies e.g., calendar information.

In general one wants to choose a model which best represents the given data. But secondly we also have to take into account the performance on feature data points. So basically when deciding to deploy such a model, one would have to generate a training data set and a test data set. The polynomial is fitted with an increasing number of k, which also increases the performance on the training set and at a certain point we will observe over-fitting. At the same time one has to evaluate the performance on the test set. This measure on the test set is also called generalization error and should have a minimum at a certain number of k. This k is then chosen for deploying and usage of the model.

Additionally one should include the knowledge about the categorical behaviours into the curve fitting, by trying to fit multiple functions to the data and also use the test set to evaluate which scenario results with the best performance.

5.2 System state clustering

As we have seen from the detailed exploratory data analysis, there is a very strong influence of the domain specific factors e.g., battery storage systems, photo-voltaic systems, heating systems and the external influence. Therefore it is of great interest how all these factors are influencing the performance and the behavior of the grid or the respective substations. It is also obvious that there can be system states which are optimal in the purpose and such states which are undesirable. Regarding the undesired system states it is also often the case that one would like to find the root causes of this behaviour. But in any case, the first step should be to identify the different system states in order to proceed further steps.

The following machine learning concept is therefore related to the **Pattern 5: different day profiles because of the environmental influences**. This unsupervised approach will demonstrate how one can extract information about the different system states without any prior knowledge or labeling. In this special case, when we refer to the system state, we are taking into account the corresponding power consumption day profiles. The next sections will introduce the concept which is based on extracting features from the daily time series followed by a clustering algorithm. After evaluating the clusters and visualization them in the time series, a polynomial regression approach is used to fit a function which can then be seen as a representative for the respective day profile class.

5.2.1 Clustering concept

Reviewing the ML design cycle from Figure 3.2 we can define the problem in this case, as finding different system states regarding their daily power consumption time series. While the data collecting for this task is already done in our data analysis steps, we can now use this preprocessed time series and apply the concept which is illustrated in Figure 5.5. Basically it can be divided into three major steps:

- Feature extraction in the spectral, temporal and statistical domain of each daily time series.
- Unsupervised clustering (k-means) in the corresponding feature space of the different days.
- Extracting the information about the frequency of the different day profiles and visualizing the clusters in the original time series.

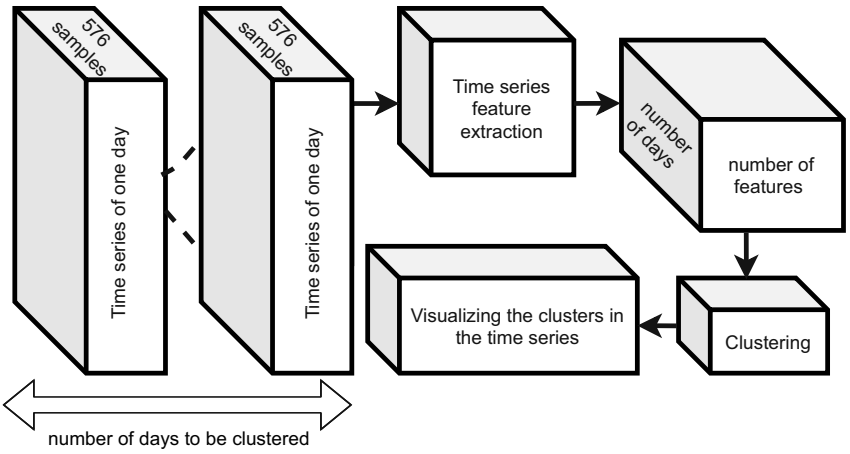

Figure 5.4: Proposed day profile clustering concept

Feature space representation

For this use case the time series is filtered and divided into windows of 24 hours, whereby midnight is always used as a boundary between two windows. The feature extraction step is now performed on each daily time series. When selecting the features, it is important to consider which information one wants to extract from the time series, as the features are representing the problem. If the features are not chosen properly, one might not be able to solve the problem, or simply produce weak results. The feature extraction library[1] which is used in this work, makes it possible to choose between the following feature domains:

[1] https://tsfel.readthedocs.io/

- Temporal: e.g., entropy, auto-correlation, peak to peak distance
- Statistical: e.g., histogram, interquartile range, maximum, minimum
- Spectral: e.g., FFT coefficients, maximum frequency

In this special case as the goal is to generate knowledge about different existing day profiles all three domains are considered for the feature extraction and clustering.

Clustering algorithm

Once we have extracted the features, which are now the representatives for each day profile we can use this information and apply an unsupervised clustering algorithm. One of the most used unsupervised clustering approaches is K-means. One of its features is, that it is quite well suited to spherical cluster shapes. But as in any case when we do apply an unsupervised approach in a highly dimension feature space, one does not have lots of prior information about the cluster shape. So one simply has to try different approaches and take the one which best solves the problem. In this case the concept is tested with different unsupervised clustering methods (K-means, hierarchical clustering, Gaussian Mixture Models) and K-means has revealed most of the information within the clusters. Therefore it is also presented and used in this work. One disadvantage of the K-means clustering is, that one has to choose the number of clusters k. In order to get a feeling for the appropriate number of clusters one can use the elbow method. This method calculates the sum of squared distances between the data points and the cluster centroids. By monitoring this measure and increasing k one will observe a shape which reminds of an elbow. The case were we reach the bend point of the elbow is simply the point were we do not significantly acquire more information with adding clusters. Therefore this can be considered as an optimal number for k.

5.2.2 Day profile curve fitting - polynomial regression

Once we have obtained the different day profile classes we can use the information to label the original time series. The labeld time series can now be used to fit a polynomial function for each time series cluster. The theoretical concept for polynomial regression is already described in section 2.2.2. By modifying and applying this supervised ML concept we can reformulate the goal towards fitting a function

$$P(t) = g(t, \mathbf{w}), \tag{5.9}$$

which describes the active power consumption dependent on the time t and the polynomial coefficients **w**. Remembering the definition of the coefficients and the features,

$$\mathbf{w} = \begin{bmatrix} w_0 & w_1 & w_2 & \dots & w_k \end{bmatrix}^T, \quad \mathbf{x} = \begin{bmatrix} 1 & t & t^2 & \dots & t^k \end{bmatrix}^T, \tag{5.10}$$

we can construct our feature matrix **X** as,

$$\mathbf{X} = \begin{bmatrix} 1 & t_{c_1,1} & t_{c_1,1}^2 & \dots & t_{c_1,1}^k \\ 1 & t_{c_1,2} & t_{c_1,2}^2 & \dots & t_{c_1,2}^k \\ . & . & . & \dots & . \\ 1 & t_{c_1,n} & t_{c_1,n}^2 & \dots & t_{c_1,n}^k \\ 1 & t_{c_2,1} & t_{c_1,1}^2 & \dots & t_{c_1,1}^k \\ . & . & . & \dots & . \\ 1 & t_{c_n,n} & t_{c_n,n}^2 & \dots & t_{c_n,n}^k \end{bmatrix} \quad \in \quad \mathbb{R}^{(n \cdot c) \times (k+1)}, \tag{5.11}$$

an the target values **y** as,

$$\mathbf{y} = \begin{bmatrix} P_{c_1}(t_1) \\ P_{c_1}(t_2) \\ . \\ P_{c_1}(t_n) \\ P_{c_2}(t_1) \\ . \\ P_{c_n}(t_n) \end{bmatrix} \quad \in \quad \mathbb{R}^{(n \cdot c) \times 1}. \tag{5.12}$$

In this case we denote c as the number of days in the investigated cluster and c_i corresponds to the respective day profile. The variable n represents the number of measurement points per day. Using this concatenated form of the different day profiles for one cluster we can obtain the optimal polynomial parameter in the following way,

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{5.13}$$

5.2.3 Results - day profile clustering at a substation with battery support

The prior introduced concept can now be applied at any substation for a chosen measurement series (e.g., active power, voltage, current) and for a desired time horizon in order to reveal day profiles with similar behaviour. As already mentioned in section 4.3.4, we do have different active power consumption day profiles because of the environmental influences or the domain specific components which are interacting with these substations. Especially substations with an installed battery storage system

do experience a strong influence on their daily consumption behaviours. While the general aim of installing such systems is to prevent load peaks, there is also presence of undesired behaviours e.g., load steps because of discharged batteries, battery maintenance events or battery failures. As such storage systems are still in development and their deployment in testbeds is rather new, it is of great interest how the system is behaving over a certain time period. It is therefore a perfect fit to investigate how many different day profiles are present in the time series. The investigated time period for this analysis is one year: 2018-01-01 – 2019-01-01.

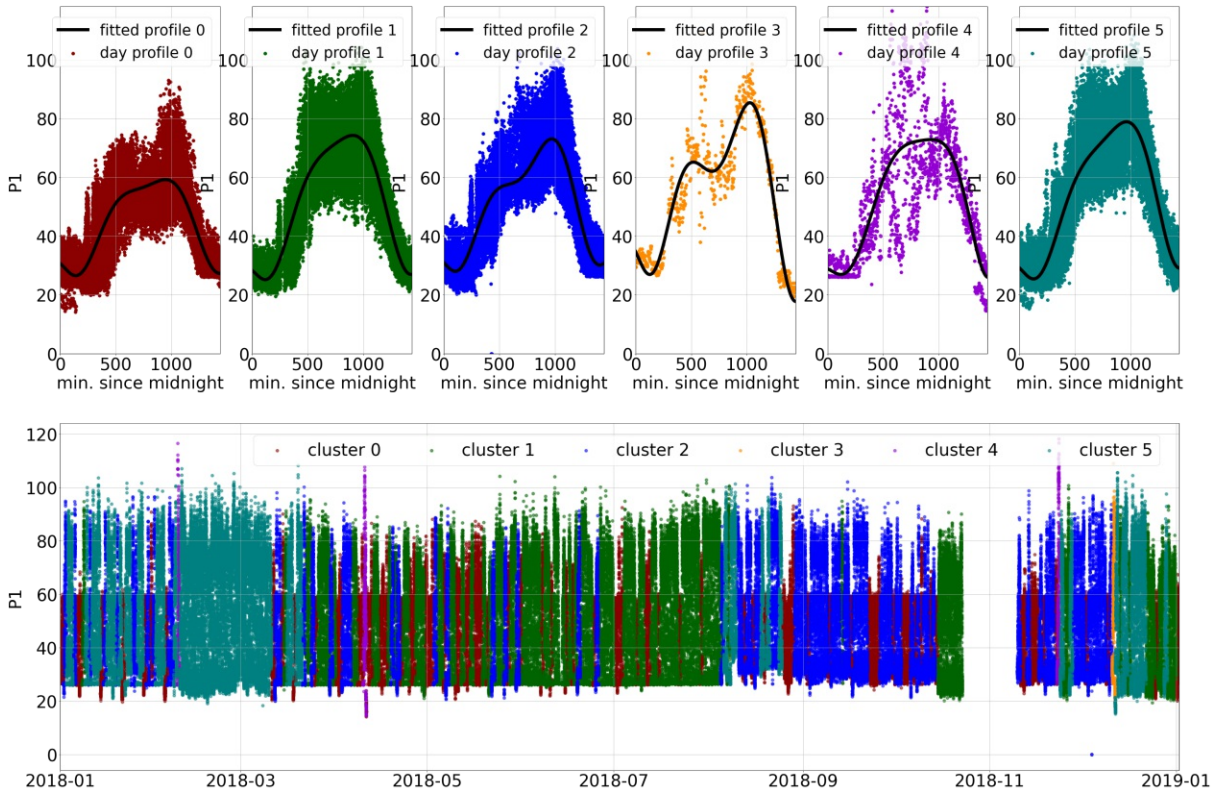

Figure 5.5: Day profile clustering result for an example subst. with an installed battery storage system

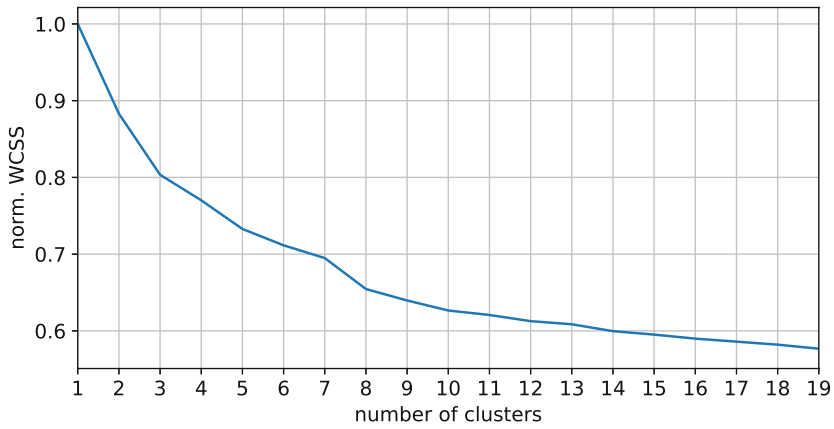

Figure 5.6: Resulting *elbow* curve for the day profile clustering example

Figure 5.5 illustrates the results for the day profile clustering approach. The number of applied clusters is determined with the help of the elbow method and results in $K = 6$. Figure 5.6 represents the normalized "Within Clusters Sum of Squares" over the number of clusters K. As can be seen it is hard to determine the exact number of clusters since the curve is monotonically decreasing and there is no unique bending point, but this method gives a good idea about the range of choosing K.
Investigating the resulting clusters in Figure 5.5 does already bring up lots of information about the general behaviour at this substation. We do observe seasonal behaviour, desired system states, undesired system states and anomalies.

Cluster 0 - The investigated substation in this case is supported by a battery storage system. The aim of this battery storage system and its control strategy refers to loading at night and preventing load peaks over $60kW$ during the day. Investigating the cluster 0 (marked in red) one will recognize that this cluster and especially its least square fit (fitted profile 0) is representing this behaviour. Taking a closer look at its appearance on the long term scale (lower part of Figure 5.5) one can see that this cluster is appearing in a periodic manner on Saturdays and Sundays. This already indicates that the battery is just able to prevent the load peaks for the whole day on weekends where there is a lower overall power consumption.

Cluster 1 and 5 - Considering the clusters 1 and 5 (green and cyan) one will recognize that this clusters are referring to day-profiles where we do not observe any influence of the battery system. The peak-shaving control in this case might simply be deactivated or subject to a system failure.
For the interested reader the question might now come up what is the actual difference between cluster 1 and 5. Since by only observing the clusters and their fitted profiles there might be no clear visible distinction. At this point we have to remember that we do cluster the day-profiles corresponding to statistical, temporal and spectral features. Using this information and observing their seasonal occurrence (lower part of Figure 5.5) we can see that the day-profiles simply might have different feature space characteristics (e.g., different rising falling slopes, different overall power consumption) which are depending on the season. In our case we can observe that cluster 1 (green) is occurring mostly in the summer and cluster 5 (cyan) is mostly occurring in spring.

Cluster 2 - Investigating the cluster 2 (blue) one will recognize that this cluster is related to the case where the battery is able to cover up the load peaks partly of the day, as in the evening hours when the battery is discharged, it again follows the original consumption profile. This is already an indicator, that our battery storage might have a too small capacity or the underlying control strategy is designed inefficiently.

Cluster 3 and 4 - Putting attention to the remaining two clusters (yellow and violet) one might already see that their behaviour looks irregular and they do have different characteristics in the feature

space. Taking a closer look at their absolute occurrence in table 5.1 we can also see that these events are quite rare. But we can relate both clusters to events which are caused by the battery system.

Observing Table 5.1, one will recognize the occurrence frequency of each cluster in absolute numbers and in %. It is definitely worth mentioning, that the system is only working in the desired system state (cluster 0) in 24.5% of the investigated time period. Cluster 2 is occurring in 24.8%. This means the peak shaving system is active in only $\approx 50\%$ of the time. The rest of the time it is inactive or in $\approx 1\%$ of the time there is the presence of anomalous behaviour. Extracting this knowledge is extremely

	cluster 0	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5
absolute occurrence	85	103	86	1	3	69
occurrence in %	24.5	29.7	24.8	0.3	0.9	19.8

Table 5.1: Example substation: occurrence of day profile clusters

valuable for the grid operator since it reveals that the battery system has to be modified in order to operate in a more optimal and efficient way. It brought up the information that the battery storage system is working very inefficient and its storage size or more realistic its control strategy has to be changed in order to ensure an optimal behaviour.

Additionally this method reveals the presence of anomalous and rare events. The knowledge about the occurrence of such events is highly important since it is the foundation of designing event detection concepts which can identify such anomalous behaviour as soon as possible in order to secure the grid and its underlying components.

5.3 Unsupervised daily mean clustering

The exploratory data analysis in chapter 4, does reveal a lot of patterns. Especially when analysing the daily mean values of the grid, there are visualizations (e.g., Figure 4.4) which often do contain different clusters. These clusters are of different shapes (spherical, oval) and separated in different ways (linear, nonlinear). Sometimes we even do not know or see the clusters, as the feature dimension space is too high and the separation is highly nonlinear. The strength of unsupervised clustering in this case is to reveal this information. But as there are many approaches as mentioned in section 2.2.1 one has to apply some important steps in order to get useful clustering results:

1. Try to visualize the distribution of the data e.g., histograms and box plots.
2. Choose a clustering algorithm which suits the distribution of the data.
3. Choose an appropriate number of clusters. In low dimensional spaces prior visualization might resolve the number of clusters. In high dimensional spaces one can use methods like the elbow method for K-means or a distance threshold in hierarchical clustering.
4. Try to visualize the clusters in different domains in order to study their root cause.

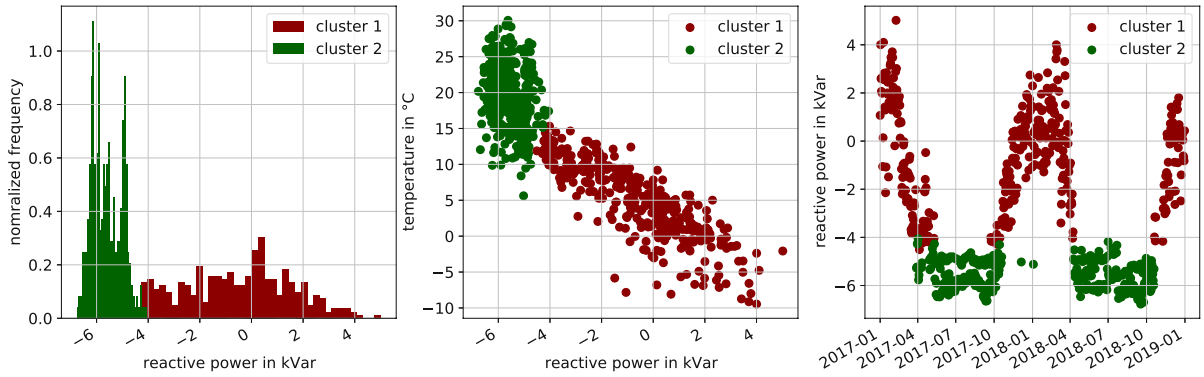

Figure 5.7: Resulting reactive power cluster with a Gaussian Mixture Model clustering

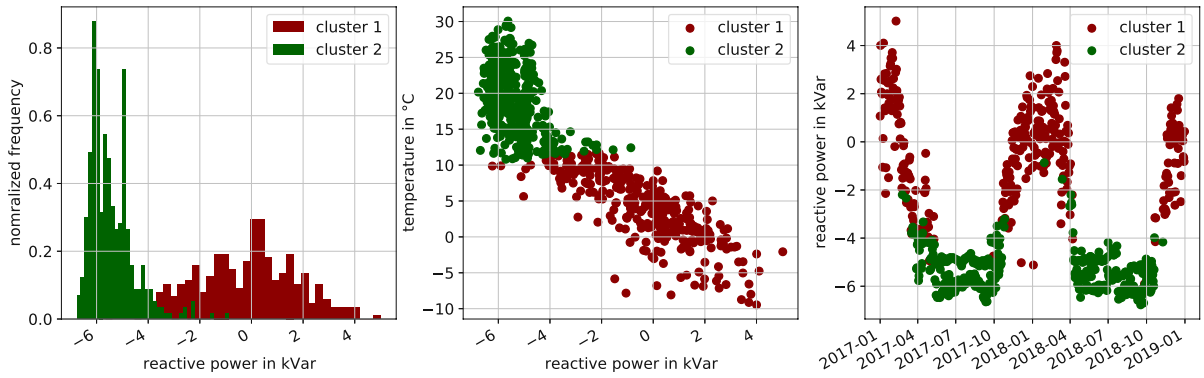

Figure 5.8: Resulting reactive power cluster with K-means clustering

Figure 5.7 illustrates one result for this unsupervised clustering pipeline evaluated at an example substation. The first step is done by visualizing the reactive power and its distribution in form of a normalized histogram (see left subplot of Figure 5.7). This histogram already reveals the information, that the clusters may come from two different Gaussian distributions. And that is already a good hint for using a Gaussian Mixture Model.

In the third step we are visualizing the two dimensional feature space. In this case it is the scatter plot between the daily mean temperature and the reactive power (see middle subplot of Figure 5.7). After visualizing this, one again recognizes the two distinctive regions, which let us define the number of clusters with K=2. The last step is trying to visualize the clustering results in a different domain. In our case we are able to map the result back to the time series domain (see right subplot of Figure 5.7).

The results in this case are quite interesting, since the Gaussian Mixture Model can identify the two different clusters well and the visualization in the daily mean reactive power time series reveals the information that the clusters are connected to a seasonal behaviour. This information about the seasonal behaviour can then be re concluded to a certain behaviour of the underlying components. In this special case we have the information about the presence of a heat pump system. This system seems to switch

its operating mode on two fixed dates during the year. Investigating the right subplot of Figure 5.7 one will identify this change points at the beginning of March (e.g., 2017-04, 2018-04) and at the beginning of October (e.g., 2017-10, 2087-10).

To also illustrate the importance of not simply using any clustering algorithm and to also verify this prior mentioned steps we can perform the same clustering task with the K-means algorithm. Figure 5.8 illustrates the result. It is visible that the spherical distance measure of the K-means algorithm is not suitable for this purpose as it does not take into account the variance of the data (see middle subplot Figure 5.8). Also crosschecking with the result in the time series and comparing it to the result of the Gaussian Mixture Model clustering (see Figure 5.7) one will agree that the two distinctive clusters are better identified by the Gaussian Mixture Model compared to K-means.

5.4 Rare event detection

Considering the results from the exploratory analysis and again reviewing the extracted patterns one will agree that we do face lots of events in the underlying data. While all these events do have different causes, or can be related to different environmental influences it is of great interest to actually detect these events. Regarding the application it might also be of advantage to detect these events as soon as possible in order to provide a more detailed and accurate information about the current grid behaviour. Since rare event detection is not an easy task and needs to be tackled to ensure the reliability of our grid, we will provide a quick introduction into general approaches to tackle the problem of detecting rare events in the next subsection. A more detailed approach which is related to simulating training data, for training the event detection algorithms, is then provided in chapter 6.

Traditional approaches of using existing data, labeling the data and then train a DL model for detecting the events is the current state of the art and rather straight forward. A more challenging task is to detect events which are unusual and rare in their occurrence. To tackle the problem of detecting such rare events is not straight forward anymore. There is simply not enough training data available in order to be able to learn the investigated behavior. Basically one can handle this challenge with to different approaches:

- **Anomaly detection** - In most of the cases there is lots of data available where the system is operating in its desired behavior. One can use this information and train an unsupervised auto-encoder in order to fit the model to the normal behaviour. Once there is the occurrence of a rare event the trained model will not be able to reconstruct this behaviour. By simply computing the difference between the input and the output one will be able to detect anomalies by investigating the reconstruction error. The advantage of this approach is that we do not need to label data for

detecting rare events. We simply have to get access to a sufficient amount of data where we are sure to not expect any irregularities or anomalies. The disadvantage on the other hand is, that we are able to detect irregularities and anomalies which are unlike the normal behaviour, but we are not able to annotate what kind of rare anomaly we are actually detecting.

- **Simulating training data** - Data augmentation but especially generating and simulating new training data can be used to generically reproduce time series behavior which corresponds to the investigated anomalies. This generic data can then be used to train the event detection models. The advantage of this approach corresponds exactly to the disadvantage of the previous method. In this case we can train the event detection on the simulated data to exactly detect these events. The disadvantage on the other hand is that we actually have to simulate the data. This does mean that we have to exactly reproduce the root causes of the real-world. This is a hard task and also incorporates a problem like distribution shift between the simulated an the real data.

Since the field of rare event detecting is of high interest, especially in the Smart Grid domain, chapter 6 is providing a concept for generating training data based on a heterogeneous multi-modal Smart Grid simulation. The concept is verified by implementing a use case which is related to generate training data for detecting a battery maintenance event.

5.5 Conclusion

Machine Learning is known for being able to solve a diverse set of problems (e.g., classifying data, structuring data, predicting data). In our case the origin, motivation and problem definition behind each ML concept is related to chapter 4, where we are actively searching for patterns. Once the knowledge about the patterns is present we decide to design ML applications in order to better understand or to make use of these patterns.

In section 5.1 we use the knowledge about the correlations between the external influence of the weather and the calendar information in order to generate a polynomial representative for forecasting the consumption based on the external influence. In the end this concept can be related to a supervised regression and predicting task. This knowledge about the power consumption prediction can be incorporated into control strategies for modern energy communities (e.g., optimizing load flows, optimizing battery storage management, optimizing heat pump control).

The second concept in section 5.2 is based on the pattern, that when observing a distribution substation and its corresponding day profiles we do face an influence to these time series because of the environmental influences (e.g., weather, calender information) and the inherent structures (e.g., battery storage, photo-voltaic system). While some of these influences are directly visible to the observer, there

is also the presence of influence which is hidden in the high resolution representation. Since the human perception is not able to detect all different system states we implemented a concept for clustering the according day profiles into a set of clusters. This concept is exploring time series behavior on a daily basis. We therefore investigate the power consumption time series for e.g., one year and extract time series features in time windows of 24 hours. Considering this feature space representation we apply an unsupervised distance based clustering algorithm to identify similar day profiles in the feature space. Each cluster is then mapped back to the time series representation and polynomial regression is used to extract a representative for each cluster. This concept is suitable to analyse the behaviour of the grid and can be used to identify the different system states. Closing the loop with the help of domain knowledge one can then identify which states are desired or undesired. In the proposed use case where we investigate a substation with a supporting battery system, we do reveal the fact that the battery is only 25% of the time performing in the desired behaviour. Therefore there is a need of changing the control strategy of the battery.

The concept in section 5.3 is trying to identify consumption clusters which are connected to the environmental influences. It provides a general approach of handling such unsupervised clustering problems and also illustrates the importance of being aware of the differences between similar clustering approaches. In the end this concept is very useful to identify seasonal behaviour e.g., changes in the control strategy of heat pumps.

The last ML concept presented in section 5.4 is related to detecting events. This sequence analysis approaches can be used for identifying events in the high resolution time series. While events with a high occurrence frequency and data basis can be detected well with supervised learning, this is not an easy task for detecting rare events. One possibility to detect rare events is to make use of anomaly detection approaches, or as also shown in chapter 6 one can use a Smart Grid simulation framework to reproduce and create training data for rare events. Detecting such rare events e.g., battery system anomalies, heat pump anomalies or system failures are of great importance to the grid operator, since some of these events might raise safety critical concerns.

Reviewing the presented ML concepts in this chapter, one might recognize that we actually use a combination of commonly known ML approaches (e.g., clustering, regression, prediction, sequence analysis) to solve the underlying task of Pattern Recognition in a heterogeneous Smart Grid environment.

Chapter 6

Rare event detection based on simulated training data

Especially in supervised learning, large training data sets are required, while often a lack of high-quality training data sets is experienced when training DL models to detect events and anomalies in the newly built heterogeneous Smart Grids. Grids for which measurement data is available are either newly build or updated. Their properties are known and they tend to be rather stable, which makes it hard to extract sufficient training data sets for rare grid events. Thus, an approach is needed to generate the data for the ML algorithms even before monitoring components are available in the grids. Especially events that are unlikely but may impose a high severity are hard to be tackled. They cannot be sufficiently tested or observed in the real environment but may be addressable if a suitable method models the cause of these events in simulations.

This chapter therefore introduces an approach to re-enact rare multi-modal grid events to generate ML training data sets using Bifrost, a heterogeneous Smart Grid simulation tool [25]. In order to tackle the problem of generating simulation data for training DL model to detect rare events we will:

A. Introduce Bifrost and its enhancements for re-enacting real-world multi-modal grid events,
B. Present a method for semi-automated data generation of realistic real-world grid events using Bifrost,
C. Verify this method by generating a training data set for a maintenance event at a battery storage system within a low voltage grid section, and
D. Train a LSTM network architecture on this simulated test set, for detecting maintenance events in real-world data.

6.1 Multi-modal simulation framework

This section is basically introducing the Multi-modal Simulation framework used in this work. It will provide an overview about the different functionalities of the used co-simulation tool Bifrost[1] and it will also propose a concept for the semi-automated data generation.

6.1.1 Bifrost core

The co-simulation framework Bifrost consists of a core simulation engine to drive dynamic data generation and a 3D web UI for the construction of settlements. The Bifrost core itself does not make assumptions as to the provenience of domain data, nor produce any. It does, however, provide a data model, which is built from a plain-text *directory*. This directory, which is freely editable even during runtime, lists syntactic (the shape of data, e.g., that a voltage consists of 3 floating-point values) and semantic (e.g., that voltage has a unit of Volt) characteristics. Within the Bifrost data model *dynamics* represent those aspects that can change (e.g., due to user interactions as in the case of a power switch, due to underlying models as in the case of houses' power consumption, or by an external simulation controller, cf. subsection 6.1.3).

External modules, connected via a REST API, can subscribe to the Bifrost data model. At every simulation loop, all registered modules are called in-order, with a payload corresponding to their subscribed data. The modules in turn can respond with modified, updated, or new dynamic values that are stored in a time-series database, and can be visualized in graphs directly on the Bifrost UI. Figure 6.1 shows the Bifrost web UI design and highlights the different heterogeneous building types. While the Bifrost core and its modules can be fully controlled via the REST API, a graphical interaction (play/pause, module configuration and result visualisation) helps the user to construct and test the individual settlement.

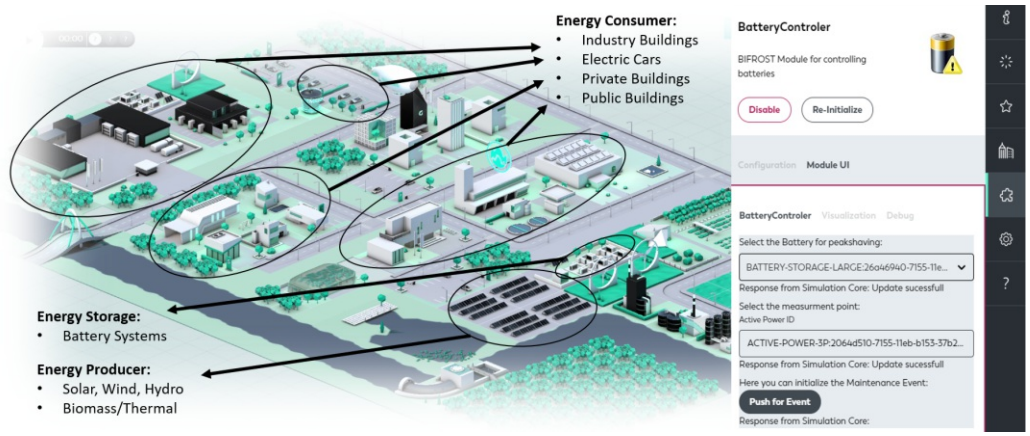

Figure 6.1: Bifrost web UI with its heterogeneous building types and an exemplary module interface [1]

[1] https://bifrost.siemens.com

6.1.2 Multi-modal Bifrost modules

The main strength of Bifrost comes with the open interface to nearly any kind of behaviour model (Bifrost *module*) and its flexibility of allowing new and diverse characteristics (Bifrost *dynamic*). While classic load flow related modules range from standard load profiles (e.g., for residential buildings) to a load flow solver, heterogeneous aspects can be introduced by modules such as a weather generator or controller modules (e.g., battery storage controller or energy community controller). Following an overview about those modules with respect to the multi-modal nature of Smart Grids, which were used for the use case presented in section 6.2, will be given. This list is not complete but limited to the presented work.

Load flow solver

This module covers the basic load flow within an electrical grid. All power values generated by other modules as well as the grid topology are analyzed and the load flow is calculated and written back to the current simulation step.

Weather generator

This module introduces various weather parameters into the simulation environment. It generates dynamics such as temperature, cloud coverage and precipitation. All values can either be taken from real historic weather data sets or can be generated based on any weather model.

Building model

The main task of the building module is to calculate the power consumption of residential and commercial buildings. It is therefore split into sub-classes, each dealing with a specific domain. The load class provides the base load consumption of the building. Those values can be extracted from standardized load profiles or specific use case related profiles. Additional effects such as randomization or noise overlay help to generate different and more realistic profiles. A photo-voltaic (PV) class handles optional solar panels on each individual building. Beside the electrical characteristics and geographical orientation, this class uses the weather data to calculate the PV output. In addition, an e-mobility (EV) class simulates the charging of an e-car based on information about the charging pole, e-car and some parameterizable characteristics. Additional features like local battery storage, or heating pumps can be added.

Battery model

This module simulates large battery storage systems, which are not part of a building but directly connected to the grid and controlled by global or communal instances. Possible applications could be the provision of (primary) control energy, local overload prevention or optimization of energy communities. The module therefore simulates a realistic behaviour of configurable batteries (e.g., by adding features such as aging and self discharge) and provides an interface for other modules, which can control the battery by sending charging and discharging commands.

Battery controller

This module is responsible for adding any kind of battery controlling strategy. Implemented algorithms include, for example, a peak-shaving method, which uses the battery to prevent local transformer overloads during the daily peak times. The main functionality used for the presented work is to re-enact a maintenance event at the battery. This event is described in detail in section 6.2.

Time controller

Although the Bifrost core handles the simulation time and step size, the additional Time Controller module is needed to manipulate the simulation time in Bifrost to target specific timestamps. This is necessary in case certain parameter settings are influenced by the current simulation date and time (e.g., weather).

6.1.3 Semi-Automated data generation

A multi-modal simulation tool such as Bifrost can now be used to create training data sets for rare multi-modal real-world grid events as a means to generate large and high-quality data sets. The proposed approach can be split into four steps, which will be introduced and described in the following subsection.

Event identification

Prior to every simulation run, the event under investigation has to be identified and analysed. This step typically involves domain experts such as grid operators and stakeholders. Once an event is identified (e.g., grid endangering weather behaviour), it has to be analysed and translated into the simulation world. Bifrost modules to re-enact a real-world event can be created in two ways:

- *Recreating the event as a time series*: If the event can be characterized by any kind of time series (e.g., specific load profile or weather period), a Bifrost module can be used to replay this time

series. Additional randomization or noise overlay can help to improve the quality of the resulting training sets.

- *Recreating the cause of the event*: If possible, instead of the event itself, the underlying cause of the event should be modeled and implemented as a Bifrost module. Thus e.g., for a battery maintenance event the controller of the battery can be modeled instead of modeling the resulting time series. Using this approach, modeling can also be done even before any real-world event was recorded. Thereby specific behavior models (e.g., physical model) or abstract model approaches can be used.

Bifrost settlement setup

In a second step a simulation environment is specified by building a settlement that contains the grid topology with different types of buildings and integrates the needed Bifrost modules (see subsection 6.1.2). This settlement should mirror the real-world situation, in which the event under investigation can occur. It is worth noting, that the setup can be iteratively optimized based on the extracted training sets and their verification.

Semi-automated simulation runs

After a settlement is specified, a single simulation run can be started and stopped via the Bifrost frontend. However, using the semi-automated simulation control tool, all simulation parameters and the event scheduler can be adapted between multiple Bifrost simulations, which are started and stopped automatically. This process is visualized in Figure 6.2.

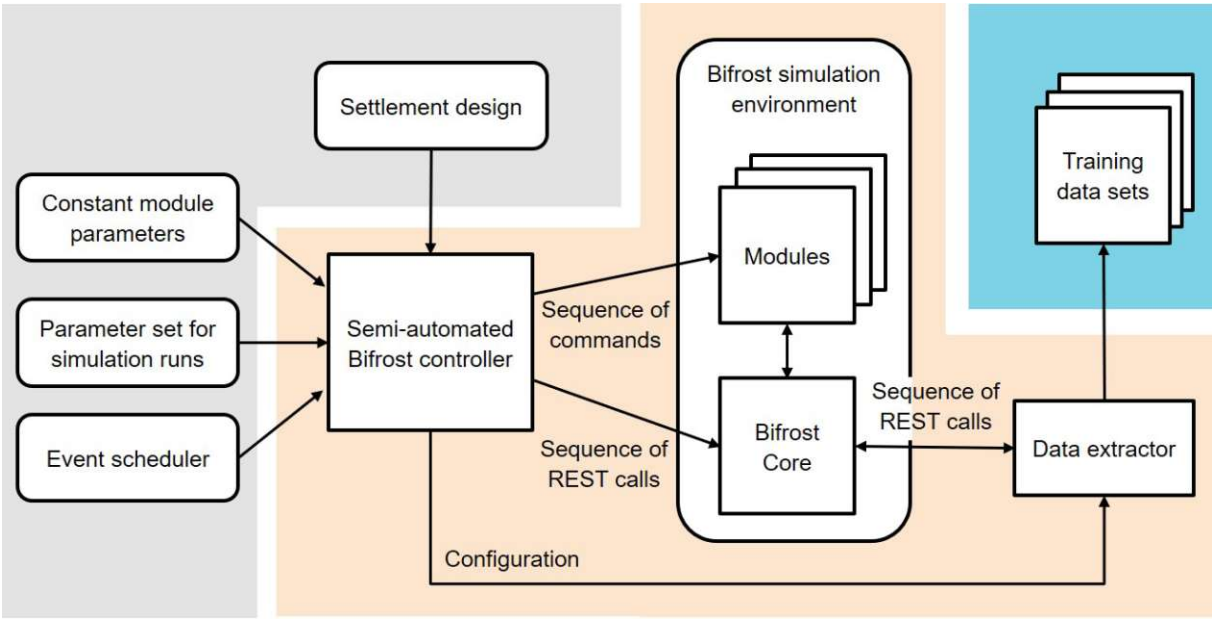

Figure 6.2: Semi-automated data generation concept

First, the entry point of the simulation has to be set according to the first two steps. This includes the settlement design and constant module parameters such as the start timestamp, simulation step resolution, chosen power profiles, as well as specific module parameters (e.g., capacity and charging power of battery storage). Next the parameter sets and an event scheduler for the event under investigation have to be defined:

- *Parameter sets*: This includes all parameters and corresponding ranges, which should be varied between the automated simulation runs. As Bifrost itself makes no assumptions about the module's purpose and function, a huge variety of modules and therefore possible parameter sets allow for diverse and realistic simulation results. For example, this could include a list of building power profiles, which should be replaced in every single simulation run. This would result in multiple runs with a different base load.
- *Event scheduler*: The event under investigation has to be triggered multiple times during the simulation run. Depending on the event and its nature, this could either be a simple schedule, which calls the event to given and maybe randomized timestamps, or a more complex model- or data-based sequence. As for all Bifrost modules, the event module designed in the prior steps can be triggered via a REST call.

After all parameters and schedules are set, the semi-automated Bifrost controller takes this information as input and generates a timed sequence of commands and REST calls, which are then automatically sent to the Bifrost core and any included module. Without further human input, multiple simulation runs are started, stopped and the resulting data sets are stored.

Training data extraction

After the automated simulation runs, the training data sets have to be extracted from the simulation results. Bifrost's data crawler module stores all simulation data (Bifrost *dynamics*) in an time-series database. The data extraction is then responsible for automatically collecting the data sets of the different simulation runs and storing them for later training of ML applications. In addition the following data processing steps are applied:

- *Data post-processing*: The simulation data is optimized for the target ML architecture. This includes, for example, manipulating the data and time resolution as well as data filtering (e.g., normalization).
- *Labeling*: Using the information from the Bifrost modules and the proposed controller enables automated data labeling. The event under investigation as well as other information (e.g., weather related events) are labeled and stored together with the data sets.

The resulting data sets can now be used according to the defined use case and ML approach. If possible, the quality of the simulation data should be verified by applying the targeted approach on historic real-word data. Experiences from such real-word tests can help to improve the training data set quality by redefining the simulation parameters and rerunning the semi-automated data generation. To illustrate the proposed data generation as well as to verify the overall concept, section 6.2 now shows the results for a battery storage maintenance use case.

6.2 Battery storage use case

The exploratory analysis in chapter 4 already revealed the strong influence of the battery storage system to the overall consumption profiles. For this use case we use time series from a single distribution substation, supported by a battery system, from 2018. The underlying battery system is configured to limit grid peaks per phase to 60 kW.

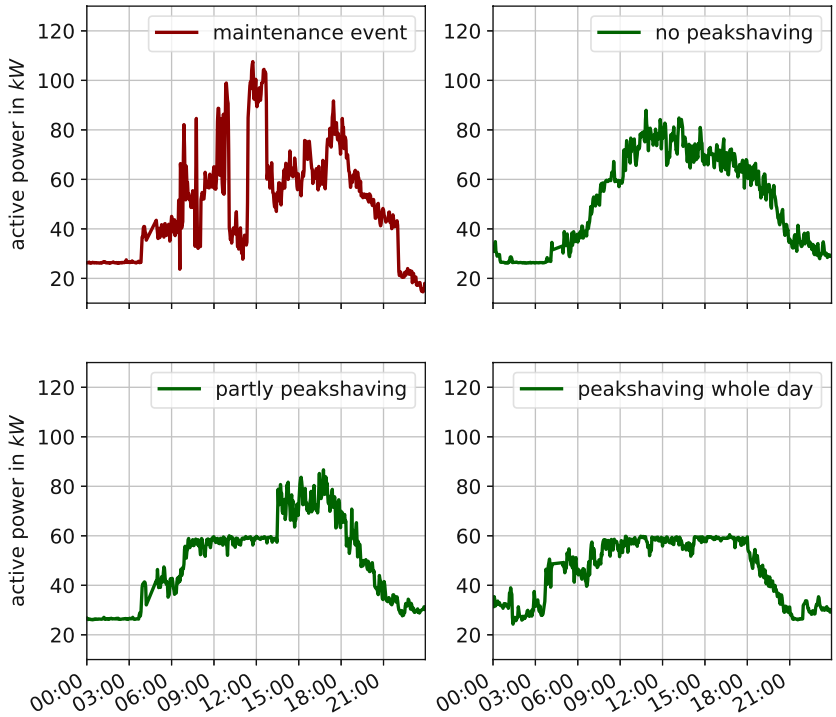

Figure 6.3: Consumption day profiles of the investigated transformer

day-profile	no peakshav.	partly peakshav.	whole day peak-shaving	maintenance event
appearance	50%	24.5%	24.5%	1%

Table 6.1: Results day profile clustering

As already derived in section 5.2 one can classify the historical time series into a certain number of day-profiles. Figure 6.3 shows the four main system behaviours:

- Battery maintenance event,
- Peak-shaving inactive over the whole day,
- Peak-shaving partly active over the day,
- Peak-shaving active for the whole day,

Table 6.1 again shows the result of the day profile clustering with the corresponding profiles and their occurrence in percentage. Depending on the use case and ML algorithm, the classification of the time series is helpful for ML training. However, the clustering also provides the information that the class "maintenance event" only occurs with 1% in this data set. To train a NN which recognizes this event in the real-world grid data, significantly more training samples are required. In order to reproduce the battery maintenance event in our multi-modal simulation, the information about the root cause and the connected parameters must be known first. In this case, the event is manually triggered on site. So time and date are random in this situation. The duration of the maintenance event depends on the storage size of the battery and the maximum charging and discharging power. These parameters are now specifically manipulated in the simulation, see 6.3.1.

6.3 Simulated training data and event detection

Taking into account the prior defined battery maintenance event, this section will introduce the results of the generated training data and the resulting event detection which is trained on the simulated data and evaluated on the real Smart grid power consumption time series.

6.3.1 Generated training data

The approach presented in subsection 6.1.3 is now used to simulate training data for the battery storage use case (section 6.2), especially to generate time series for identifying the battery maintenance event in the historic data. For this the used Bifrost settlement is designed to behave in a similar way like the testbed Aspern[1], and in order to create a diverse training set the parameters of the Building Model, Battery Module and Battery Controller (e.g. load profiles, event start time, charging/discharging power of the battery, see subsection 6.1.2) are modified between simulation runs.

Figure 6.4 illustrates the created/simulated normalized training data $p1$, where the battery maintenance event is at least triggered once per day. The second half of this graph contains two example day profiles. The corresponding labels are automatically extracted from the simulation and mark the discharging and charging period of the battery.

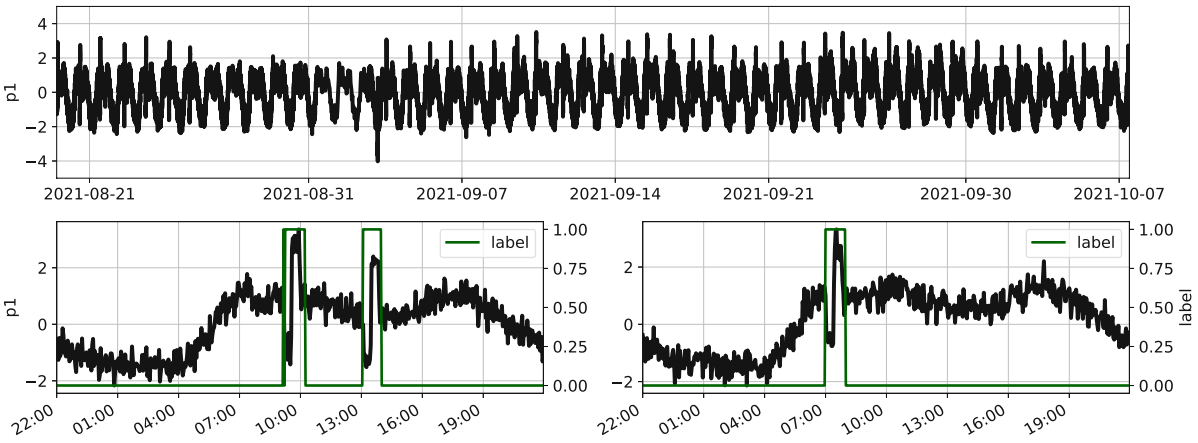

Figure 6.4: Simulated data set

6.3.2 Machine Learning concept - Neural Network architecture

When it comes to analyzing/classifying time series, LSTM networks are frequently used [26]. Their ability to learn patterns in a sequence seems almost perfect for this battery maintenance use case. In this application the Keras framework[2] and its LSTM implementation is used. Table 6.2 lists the implemented layers as well as those parameters, that differ from the standard implementation. Additionally a dropout layer (dropout rate of 0.1) is inserted between the individual layers in order to avoid over fitting.

layer	L1: LSTM	L2: LSTM	L3: Dense
parameter	units = 264 rec.act. = “tanh”	units = 64 rec.act. = “tanh”	act. = “tanh”

Table 6.2: Neural Network architecture [1]

The network is trained with a sequence length of 25 (which corresponds to approx. 1 hour with a sampling rate of 2.5 minutes). The input to the network is defined as $\mathbf{X}_{train} = [\mathbf{p1}, \dot{\mathbf{p1}}, sin(\omega \mathbf{t})] \in \mathbb{R}^{25\times 3}$, where $\mathbf{p1}$ is the normalized (zero mean, unit standard deviation) time series of the power consumption and $\dot{\mathbf{p1}}$ its derivative. To let the network also recognize/learn relative temporal relationships, the third feature vector consists of the relative time $sin(\omega \mathbf{t})$, where $\omega = \frac{2\pi}{24 \cdot 60 \cdot 60} \frac{rad}{s}$.

6.3.3 Event classification

After training the previously presented LSTM network with 30 *epochs* and a *batch size* of 200 samples, a training accuracy of 98% is achieved. This shows that the network is capable of detecting the event in the simulated data. However, more interesting are the results on the real grid data. Figure 6.5 shows the prediction of the network on the real grid time series of 2018 (with a recording gap due to

[2] https://github.com/keras-team/keras

malfunctioning sensors during autumn). One of the first conclusions which can be made is that the network detects only 5 events, which is satisfactory when crosschecking the result of the day profile clustering (battery maintenance events appear 1% of the time).

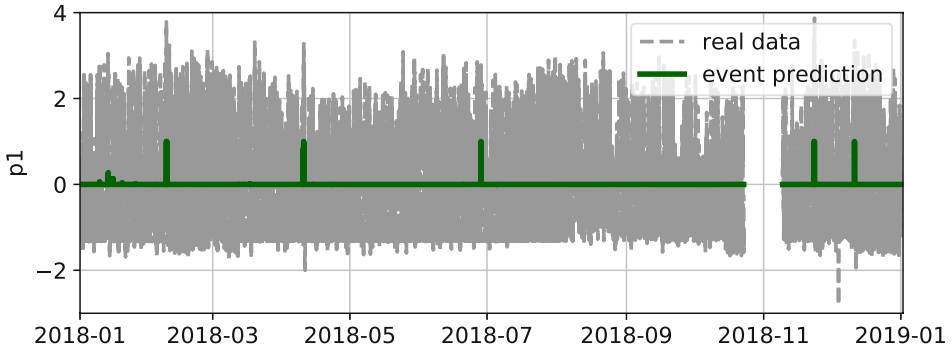

Figure 6.5: Overall battery maintenance event classification

Figure 6.6 illustrates four detailed day profiles and their battery maintenance event prediction. Apparently in all four cases anomalous behavior is prevalent and the battery can be the reason for this behavior. However, the main attention should be drawn to the two day profiles on the right hand side of the figure. In 2018-04-10 the maintenance event is detected in addition to two other anomalies. In 2018-11-23 the event is carried out twice in a row and is detected perfectly on both occasions.

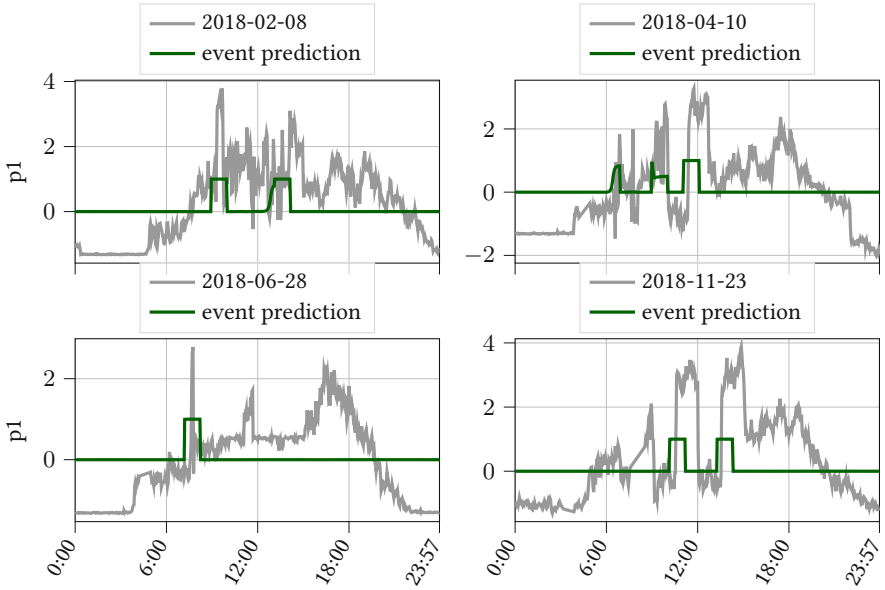

Figure 6.6: Daily based battery maintenance event classification

6.4 Conclusion

In this chapter we demonstrated a concept for re-enacting rare multi-modal real-world grid events to generate training data sets for ML algorithms. The approach is based on the heterogeneous simulation tool Bifrost as well as a semi-automated simulation controller.

The approach is verified by re-enacting a battery maintenance event within a low voltage grid section. The results show that it is possible to generate suitable training data sets for this event and to use them to train a LSTM network architecture, which is able to detect the maintenance event within real-world data from a Smart Grid testbed.

The main challenges lie in the semi-automated simulation configuration as well as the parametrization with respect to the ML algorithm since the simulation configuration is of great importance for the quality of the results. Although the used modules already provide a heterogeneous and realistic environment for many use cases, domain experts and knowledge have to be included into the configuration process for the event under investigation.

Chapter 7

Conclusion

Observing a modern Smart Grid and just considering the sampled grid measurements is an old-fashioned and outdated way of looking at this highly dynamical and heterogeneous system. Since Smart Grids are nowadays a source of energy and information, there is a strong need for actually analysing and optimizing their behaviour. In order to be able to represent the overall characteristics of a Smart Grid, it is important to include the diverse set of **energy consumers** (eg., industry building, public buildings, private buildings), **energy producers** (e.g, renewable, power plants), **energy storages** and **environmental influences** (e.g., weather, seasonal information).
Starting at such an abstract viewpoint is very important for fully understanding and analysing Smart Grids, but does also incorporate lots of challenges (see section 1.2). Especially the lack of the analysis and the extraction of patterns related to the environmental and heterogeneous influences is very prominent. These patterns (e.g., seasonal, desired or undesired behaviour of components) should be used to optimize the reliability and efficiency of Smart Grids. The next section is therefore summarizing the proposed methodology which is mainly consisting of two parts: a pipeline for extracting patterns and a design cycle for developing Machine Learning concepts based on the patterns.

7.1 Approach and results

Arriving at the task of analysing and investigating the behaviour of a heterogeneous Smart Grid environment, it is of great importance to apply a straight methodology. Otherwise the probability is high to get lost in small details. In order to keep the focus on the main goals, we propose a methodology (see chapter 3) which serves as a guideline for Pattern Recognition in a heterogeneous Smart Grid environment. It basically consists of two main steps:

- **Extracting patterns from a heterogeneous Smart Grid**
- **Defining Machine Learning concepts based on patterns**

The first step has shown that by designing a pattern extraction pipeline we are able to incorporate the heterogeneous Smart Grid environment into the process of extracting knowledge and information. This concept of extracting patterns is then verified with the help of historical data of a real heterogeneous Smart Grid environment in the testbed Aspern in Vienna, Austria. The environmental influences in this case are limited to the availability of temperature, solar radiation and calendar information. After performing the step of analysing each of the 12 distribution substations, with the help of different visualizations, animations and correlations we do reveal a set of important and interesting patterns e.g., different day profiles caused by heat pumps or batteries, correlations between grid measurements and environmental influences and anomalies related to a battery storage system. A detailed listing of all the extracted patterns and the overall results of the pattern extraction pipeline can be found in chapter 4. The main strength of the proposed pattern extraction pipeline is its modular approach. Each block can be modified on its own, without the need of changing the functionality of other blocks. This is especially interesting when thinking about comparing different Smart Grids with each other.

Once we have actually identified patterns which are representing a certain behaviour we want to use this information for defining Machine Learning concepts. All the developed concepts can be observed in chapter 5, whereas each concept is dedicated to one or a subset of patterns. In the end the tricky and most crucial part is to actually define a suitable problem which can be tackled with Machine Learning. In this thesis we basically focus on three main problems which are solved with a corresponding concept.

The first one solves the problem of **forecasting** the energy consumption based on the heterogeneous data (e.g. calendar information and temperature). This concept is realized with supervised polynomial regression. Its main advantage is to incorporate as much heterogeneous influence as possible into the forecasting. This results in a strong increase in the prediction accuracy. The predicted power consumption is of high value since it can be used to optimize load flow controllers (e.g., control strategies for batteries, heat pumps, cooling systems).

The strength of the second concept is to investigate the hidden information in the daily time series of a distribution substation. The crucial part is to actually gain the knowledge about the shape and the amount of the different system states. In this case the problem is solved with calculating timeseries features for each day profile. Unsupervised **clustering** is then used to identify groups of similar time series behaviour and the identified clusters are then analyzed according to their appearance frequency and again visualized in the original time series representation. It is worth mentioning, that this concept does reveal highly important information about the desired and undesired behaviour of a battery storage system. The analysis shows that the system is only performing 25% of the time in the desired system state of preventing load peaks for the whole day. During the rest of the investigated time period,

we do face day profiles/system states which are corresponding to undesired states. So this concept is ideally suited for identifying different system conditions. Once the domain expert has then annotated these conditions into desired and undesired, one can change parameter of the system and then again perform the same analysis and validate if the condition is now classified as a desired behaviour.
The third Machine Learning concept is related to **event detection** and its main focus lies on detecting rare Smart Grid events. Since Deep Learning methods do rely on a high amount of training data we introduce a concept for re-enacting rare multi-modal Smart Grid events in order to generate high quality training data. The concept and a dedicated result, which presents simulated training data and a Neural Network Architecture for detecting a battery maintenance event, is provided in chapter 6. The main benefit of this concept is the multi-modal structure of the simulation environment and the capability of generating Machine Learning training data which incorporates the heterogeneous nature of the Smart Grid. It is also of high interest for the grid operator to incorporate such event detection algorithms into the process of decision making regarding faults and failures. Providing algorithms which can assure that anomalous behaviour is connected to a rare but verified event can help the decision maker to decide if a certain behaviour is safety critical or just an already known irregularity.

7.2 Limits and outlook

Considering an abstract and heterogeneous starting point for the investigation of the behaviour of a Smart Grid sounds like a reasonable approach, but still does require lots of prior assumptions. One of the main constrains is simply the availability of the heterogeneous data sets. Extracting patterns and correlations between external influences and the grid measurements is therefore only possible if we do have access to the corresponding data. So one of the future steps should be to incorporate the recording of the heterogeneous measurements at each distribution substation.

This thesis does reveal lots of different patterns, but one of the main drawback is, that most of the patterns are identified with human perception. So the human in the loop has to investigate and interpret the results of the data analysis. This human factor can also be seen as a strong bias which is influencing the results. One of the main challenges is to avoid this bias and to try to implement a modified pattern extraction concept which is less dependent on the human factor and more focused on learning an detecting structures with the help of algorithms. So the question which still needs to be answered is how we can automate the pattern extraction to also uncover patterns which are hidden in the bias of our visual perception.

Trying to transfer this knowledge to the level of the grid operators might be the most difficult task. All the developed methods and approaches for extracting patterns and using Machine Learning concepts for optimizing and analysing the Smart Grid behavior, just have a high value for the gird operator if we are able to transfer this useful knowledge in a language which can be understood by them. Therefore there is a strong need of interaction between the data scientists and the grid operators in order to identify which of the problems are really worth to be tackled. Data scientists are great in uncovering hidden information and handling machines with a variety of algorithms and Machine Learning concepts, but sometimes this is just possible if they are also having access to the corresponding domain knowledge from experts. In the end there is a strong need of strengthening the interaction between domain experts and machines. One research discipline which is also connected to this topic is Explainable Artificial Intelligence, where humans are included into the reasoning and decision making of Machine Learning algorithms. This research field can form a bridge between the very complex and nontransparent Machine Learning environment and the applications handled by the grid operators.

Bibliography

[1] D. Hauer, M. Bittner, S. Cejka, R. Mosshammer, F. Kintzler, T. Leopold, and S. Wilker, "Re-enacting rare multi-modal real-world grid events to generate ml training data sets," in *IEEE International Symposium on Industrial Electronics*, to be published 2021.

[2] Y. Zhang, T. Huang, and E. F. Bompard, "Big data analytics in smart grids: a review," *Energy Informatics*, vol. 1, no. 1, p. 8, Aug. 2018. [Online]. Available: https://doi.org/10.1186/s42162-018-0007-5

[3] A. Sanchez and W. Rivera, "Big Data Analysis and Visualization for the Smart Grid," in *2017 IEEE International Congress on Big Data (BigData Congress)*, Jun. 2017, pp. 414–418.

[4] S. Sagiroglu, R. Terzi, Y. Canbay, and I. Colak, "Big data issues in smart grid systems," in *2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA)*, Nov. 2016, pp. 1007–1012.

[5] E. Hossain, I. Khan, F. Un-Noor, S. S. Sikander, and M. S. H. Sunny, "Application of Big Data and Machine Learning in Smart Grid, and Associated Security Concerns: A Review," *IEEE Access*, vol. 7, pp. 13 960–13 988, 2019, conference Name: IEEE Access.

[6] R. Hossain, A. Maung, A. Than Oo, and S. Ali, *Smart Grid*, 11 2013, vol. 132.

[7] M. Shabanzadeh and M. Moghaddam, "What is the Smart Grid? Definitions, Perspectives, and Ultimate Goals," Nov. 2013.

[8] M. Mohammadi and A. Al-Fuqaha, "Enabling Cognitive Smart Cities Using Big Data and Machine Learning: Approaches and Challenges," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 94–101, Feb. 2018, conference Name: IEEE Communications Magazine.

[9] M. Farmanbar, K. Parham, O. Arild, and C. Rong, "A Widespread Review of Smart Grids Towards Smart Cities," *Energies*, vol. 12, p. 4484, Nov. 2019.

[10] D. Zhang, X. Han, and C. Deng, "Review on the research and practice of deep learning and reinforcement learning in smart grids," *CSEE Journal of Power and Energy Systems*, vol. 4, no. 3, pp. 362–370, Sep. 2018, conference Name: CSEE Journal of Power and Energy Systems.

[11] T. S. Bomfim, "Evolution of machine learning in smart grids," in *2020 IEEE 8th International Conference on Smart Energy Grid Engineering (SEGE)*, 2020, pp. 82–87.

[12] A. M. Foley, P. G. Leahy, A. Marvuglia, and E. J. McKeogh, "Current methods and advances in forecasting of wind power generation," *Renewable Energy*, vol. 37, no. 1, pp. 1–8, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0960148111002850

[13] D. Lee and R. Baldick, "Short-term wind power ensemble prediction based on gaussian processes and neural networks," *IEEE Transactions on Smart Grid*, vol. 5, no. 1, pp. 501–510, 2014.

[14] A. M. Foley, P. G. Leahy, A. Marvuglia, and E. J. McKeogh, "Current methods and advances in forecasting of wind power generation," *Renewable Energy*, vol. 37, no. 1, pp. 1–8, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0960148111002850

[15] H.-T. Yang, C.-M. Huang, Y.-C. Huang, and Y.-S. Pai, "A weather-based hybrid method for 1-day ahead hourly forecasting of pv power output," *IEEE Transactions on Sustainable Energy*, vol. 5, no. 3, pp. 917–926, 2014.

[16] C. M. Bishop, *Neural Networks for Pattern Recognition*. USA: Oxford University Press, Inc., 1995.

[17] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[18] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh, *Feature Extraction: Foundations and Applications (Studies in Fuzziness and Soft Computing)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[19] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[20] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[21] J. Schmidhuber, "Deep learning in neural networks: An overview," *CoRR*, vol. abs/1404.7828, 2014. [Online]. Available: http://arxiv.org/abs/1404.7828

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[23] M. Deisenroth, A. Faisal, and C. Ong, *Mathematics for Machine Learning*. Cambridge University Press, 2020.

[24] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*, 2020, https://d2l.ai.

[25] R. Mosshammer, K. Diwold, A. Einfalt, J. Schwarz, and B. Zehrfeldt, "Bifrost: A smart city planning and simulation tool," in *Intelligent Human Systems Integration 2019*, W. Karwowski and T. Ahram, Eds. Cham: Springer International Publishing, 2019, pp. 217–222.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.