DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Informatics

Appliance Identification Benchmark

Jonas Buchberger

DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Informatics

Appliance Identification Benchmark

Geräteidentifikations Benchmark

Author:	Jonas Buchberger
Supervisor:	Prof. Dr. rer. pol. Hans-Arno Jacobsen
Advisor:	M.Sc. Daniel Jorde
Submission Date:	15.03.2019

I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

Garching, Jonas Buchberger

Acknowledgements

The author of this paper would like to especially thank M.Sc. Daniel Jorde, the supervisor of this paper. Without his time investment through every step of the thesis, this paper could not have been accomplished. Firstly, I would like to appreciate his quick support, feedback and help with each problem that occurred during the process. Secondly, the interesting thesis topic he offered me allowed me to continue researching on a topic I have already worked with for two years. Also, I am particularly grateful for his assistance and flexibility.

Further, I would like to thank M.Sc. Anwar Ul Haq, my former seminar supervisor, for helping me to find a suitable supervisor for my preferred thesis topic and bringing me into contact with M.Sc. Daniel Jorde.

Special thanks also to the Technical University Munich and especially to the chair of Business Information Systems and its head Prof. Dr. rer. pol. Hans-Arno Jacobsen for giving me the opportunity to write an interesting thesis and supporting me with the necessary infrastructure.

I would also like to express my gratitude to Dipl.-Ing. Thomas Kriechbaumer, Christoph Doblander, Dipl.-Inf. (FH) Matthias Kahl and M.Sc. Pezhman Nasirifard for leading the seminars that introduced me to Non-Intrusive Load Monitoring and gave me the inspiration for the thesis topic.

I would like to dedicate this work to my beloved parents, Willi and Ingrid, and my sister Lisa who supported me through each stage of my life and were always there for me when I needed them.

Abstract

Monitoring electrical loads gains more and more interest in consumer or industrial environments due to climate change and other energy based restrictions. Therefore, Non-Intrusive Load Monitoring (NILM) tries to provide cheap methods for disaggregation combined loads measured by smart meters. This includes the development and evaluation of different features in combination with machine learning algorithms. The goal is to achieve an computationally appropriate algorithm with nearly no incorrectly classified appliances. There are two algorithms presented in this paper and their disaggregation performance is evaluated on a sample dataset. Both approaches are based on classifying events, generated by appliance state transitions. These events include current and voltage measurements, sampled with a rate of multiple kHz. The first algorithm is based on Extra Trees and utilizes the Device Current Signature (DCS) of appliances as feature. The DCS is formed by combining the current peaks of event windows and using steady and transient state information of the load curves. The second algorithm uses Principal Component Analysis (PCA) on the Apparent Power S, Active Power P, Reactive Power Q and Distortion Power D (SPQD) trajectories for classifying appliances. The necessary data for testing is provided by the Building-Level fUlly-labeled Dataset (BLUED). The first algorithm achieves the best results with a F_1-Score of up to 96%. The second algorithm does not perform as well, with a F_1-Score of only 80%.

Contents

1 Introduction

Currently, the global energy market undergoes rapid changes due to global warming, the ensuing energy restriction and the massive roll out of smart homes and smart devices. The upcoming demand on analysis of electrical systems and energy reduction motivated many scientist to perform studies based on monitoring loads. Especially appliance-level consumption and activity monitoring draws the attention of multiple researchers. Appliance Load Monitoring (ALM) tries to provide the consumer with detailed household energy information about individual appliances. This way, the consumer is granted the opportunity to lower their power consumption by turning of unused devices or purchasing more energy efficient ones. ALM is not only useful and applicable for energy observation in households, but also in industrial environments. The recorded data can be used by energy companies to understand device usage patterns of their costumers and optimize their power distribution according to the needs of the consumers. Also, there is the opportunity to use ALM for power grid fault detection [48]. For collecting the detailed energy data, there are mainly two methods, Intrusive Load Monitoring (ILM) and Non-Intrusive Load Monitoring (NILM). The ordinary Intrusive Load Monitoring approach, which means equipping every device with a measurement unit on an appliance-based level, is very time-consuming and cost-intensive. This can also lead to privacy issues depending on your measurement area. Non-Intrusive Load Monitoring provides a cheaper method to measure electrical loads on an appliance-based level with less privacy concerns. The idea is to use a single smart meter to gather the aggregated mains for all appliances in an area. This aggregated data can then be disaggregated using machine learning algorithms. Researches try to get the consumption information of individual devices within the measured circuit with a proper detection accuracy and performance [6]. Therefore, various amounts of disaggregation algorithms for identifying appliances in aggregated data were already developed over the past two decades. Many of them need to evaluated and compared under different situations to confirm their validity and accuracy.

In this regard, two algorithms, presented in [1] and [15], are recreated and tested under different conditions. Both algorithms utilize event-based classification on high frequency measurements of voltage and current. The first algorithms is based on Extra Trees and uses the Device Current Signature (DCS) of appliances for classification. The algorithm combines the steady and transient information of load waves together

with the peaks in each cycle of the measurement window. The second algorithm uses Principal Component Analysis (PCA) on Apparent Power S, Active Power P, Reactive Power Q and Distortion Power D (SPQD) trajectories to create a non-overlapping feature space for the different appliance classes. In the extended evaluation of each algorithm, it is attempted to increase the disaggregation accuracy by combining multiple features presented in a NILM feature study [19]. All tests are performed on the Building-Level fUlly-labeled Dataset (BLUED) [2].

The rest of the paper is structured as follows: Chapter 2 presents related work in connection with disaggregation algorithms based on NILM. The following chapter, Chapter 3, gives an overview of NILM methods and terms. It also states some problems that occurred in the NILM society. In Chapter 4 the dataset is introduced and the preprocessing applied to the dataset is presented. Chapter 5 states the first NILM algorithm and its evaluation. The Following chapter, Chapter 6, does the same for the second algorithm. Further research and evaluation ideas are presented in Chapter 7. This paper is concluded by a short summary and a comparison of both algorithms in Chapter 8.

2 Related Work

Appliance identification is the major part of Non-Intrusive Load Monitoring and is usually modeled as a machine learning problem. First algorithms were developed in the 1980s with the beginning of the NILM idea. Basically, NILM algorithms can be divided in event-based and non-event based algorithms. The event-based approach extracts a window around an appliance event and takes this information for classification. In contrast, the non-event based counterpart takes all measured data points into consideration. Most algorithms use precomputed feature spaces and try to create an appliance specific model which is as little overlapping to other appliance classes as possible. Apparent Power, Reactive Power and Real Power are most commonly used as features. The NILM pioneer George Hart was the first to develop more appliance specific signatures using harmonics and transients of load waves for disaggregation [12]. In 1995, Leeb et al. [31] implemented spectral analysis using wavelet transformation, trying to create a model to cover all four appliance types. Using V-I trajectories started in 2005 with Ting et al. [43] and Later in 2005 with Lam et al. [30]. A collection of current state of the art and newly developed features for NILM algorithms was created by Kahl et al. in 2017 [19]. In this paper, all features were tested with multiple classifiers: Linear Discriminant Analysis, K Nearest Neighbours, Support Vector Machines and Boosted Decision Trees. Also, most of the commonly used datasets for NILM purposes were used: PLAID [10], WHITED [20], BLUED [2] and UK-DALE [23].

2.1 Supervised Learning

Hart uses a simple clustering based approach using features of the P-Q plane. Due to its simplicity, the algorithm is not suitable for all four appliance types with overlapping feature spaces [12] . These problems are later addressed by multiple researchers [7, 4, 3].

Besides the frequently used Hidden Markov Models (HMM) [47, 26, 46, 35], Support Vector Machines (SVM) [8, 19], k-Nearest Neighbours (KNN) [19] and Artificial Neural Networks (ANN) [40, 18, 22] are more and more applied on NILM problems. ANNs especially benefit from incorporating temporal as well as appliance state transition information in their learning [48]. HMMs also share this advantage, but their major drawback is the exponential increase of computation cost when adding more appliances.

In addition, the model can not be extended and has to be retrained each time a new appliance is added. Hence, ANNs offer a higher flexibility and expandability.

SVMs, however, show good results using harmonic signatures and low frequency features [32, 33, 21]. A hybrid model using SVM and GMM is proposed by Lai et al. [29].

In 2017, Kumar et al. proposed an algorithm based on Graph Signal Processing (GSP) of low frequency data [28], showing promising result on the REDD [25]. Recently in 2018, Kanghang He et al. also proposed algorithms based on GSP [13]. The first approach is based on total graph variation minimization. The second one uses the total graph variation minimization as starting point for simulated annealing. Both algorithms were tested on two real household datasets, showing competing results to the traditionally used HMMs or Decision Trees.

2.2 Unsupervised Learning

Goncalves et al. [37] presented a method to use the delta of steady P and Q features for extracting appliance information form the aggregated load waves. They use Genetic K-means and agglomerative clustering to determine the number of appliance classes in an unsupervised way. Each cluster is then seen as a combination of multiple overlapping load waves which are further broken down into their individual signals. Next, the algorithm tries to minimize the distance between an unknown sample and an already existing cluster. The author claims that especially low powered devices such as lamps with similar load signatures produce a high number of false positives. In addition, multi-state appliances are difficult to reconstruct due to various clusters formed by the different appliance states. In contrast, appliances with higher power consumption form their own unique cluster and can therefore be easily detected.

Also, non-powered features are taken into consideration for energy disaggregation. Kim et al. create appliance specific Hidden Markov Models (HMM) utilizing the usage duration and daytime of appliances along with their real power signature [24]. In comparison with other variations of HMMs, Conditional Factorial Hidden Semi-Markov Models (CFHSMM) show the best performance in unsupervised algorithms with an accuracy of 83%. The addition of non-powered features improved the disaggregation accuracy. Adding more appliances, however, decreases the model's performance, as stated by Zoha and Kim [48, 24].

An other HMM based approach is presented by Pattem [38]. Pattem uses the abrupt transitions or magnitude changes of the power trajectories as features for an unsupervised disaggregation algorithm. In addition, a novel approach of the Viterbi algorithm for sequence decoding is used and special focus is put on handling low

powered appliances. The algorithm results in an effective method for magnitude-based disaggregation.

More variations of Markov Models, called Additive Factorial Approximate MAP (AFMAP) and Hierarchical Dirichlet Process Hidden Semi Markov Model (HDP-HSMM), were used by Kolter and Johnson [26, 16].

Heracleous et al. applied Conditional Random Fields (CRF) on real-world energy data. The algorithm uses clustering methods and histogram analysis to detect the ON/OFF states of appliances. They claim that the usage of CRF has a significant performance increase in comparison to HMM. The data of 21 households were used for a binary classification experiment, resulting in an accuracy of 86% for CRF and a much lower accuracy for HMM [14].

3 Non-Intrusive Load Monitoring

With the upcoming demand on analysis of electrical systems and energy reduction, cheaper methods for measuring and monitoring loads have gained more importance over the years. Appliance Load Monitoring (ALM) tries to provide the consumer with detailed household energy information about individual appliances. Therefore, the Non-Intrusive Load Monitoring (NILM) method utilizes the aggregated energy data collected by a single smart meter. The aggregated data can then be disaggregated to get further information about individual devices in the circuit. There are various amounts of approaches of disaggregation algorithms for identifying appliances in aggregated data. Many of them need to be evaluated and compared under different situations to confirm their validity and accuracy. Figure 3.1 shows the different approaches and steps of appliance identification that are commonly used.

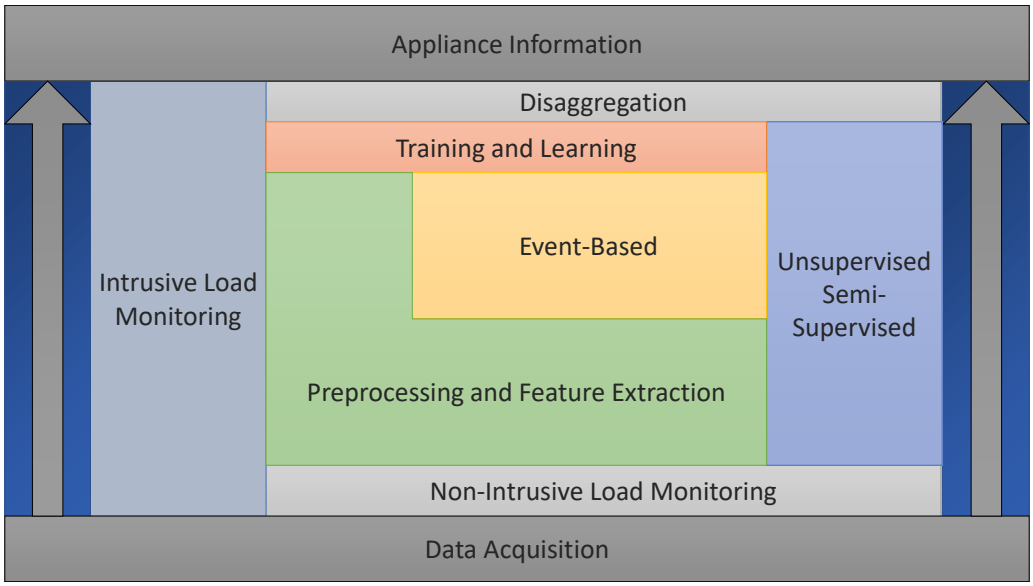

Figure 3.1: Different approaches monitoring loads.

3.1 Data Acquisition

Data acquisition is the first part of the NILM process. The current and voltage waves have to be measured and recorded. Here is also the first separation of NILM algorithms. The data can either be recorded with an high sampling rate with up to 1000kHz or a low sampling rate of 1Hz or lower [48]. A higher sampling rate provides more detailed information for the disaggregation process, but it is also more costly and storage-intensive. For example, the Building-Level Office eNvironment Dataset (BLOND) is one of the largest recorded datasets with high sampling rates and open source existing. It contains two datasets: one with 250 kHz aggregated mains and 50 kHz ground truth data recorded over 50 days and a second one with 50 kHz aggregated mains and 6.4 kHz ground truth data recorded over 213 days. These datasets were recorded for NILM testing purposes at the Technical University of Munich [27]. A lower sampling rate is more cost and storage efficient but results in a higher loss of information which can lower the disaggregation accuracy. Because of the big differences in the data acquisition, the NILM algorithms can be separated into low and high frequency ones [48].

3.2 Event Detection

An event is defined as a change in the current load wave that occurs when an appliance switches its state. Detecting these events is one of the major parts in current disaggregation algorithms for NILM approaches. Appliances are separated into multiple categories. This information can be used as features for disaggregation. The common consumer appliances can be divided into four categories. The ON/OFF State Devices, the Finite State Machines or Multiple State Devices, the Continuously Variable Devices and the Permanent Consumer Devices. In order to categorize load curves to these four types, steady-state and transient event based feature extraction algorithms have been developed. The steady-state method can be realized with a low cost hardware, whereas the transient event detection needs better hardware. However, it also offers more detailed energy information to use for the load identification algorithms. Moreover, methods are developed to skip event detection and just use the raw load curves [48]. Examples for the different appliances types and their theoretical load curves can be seen in figure 3.2 and 3.3. The fan heater and the oven belong to the Single-State Devices (Type I), the blender and the workstation belong to the Continuously Varying Appliances (Type III) and the stove burner is an example for a Multi-State Device or Finite State Machine (Type II). Type IV examples are telephone sets or hardwired smoke detectors which usually have a constant power draw.

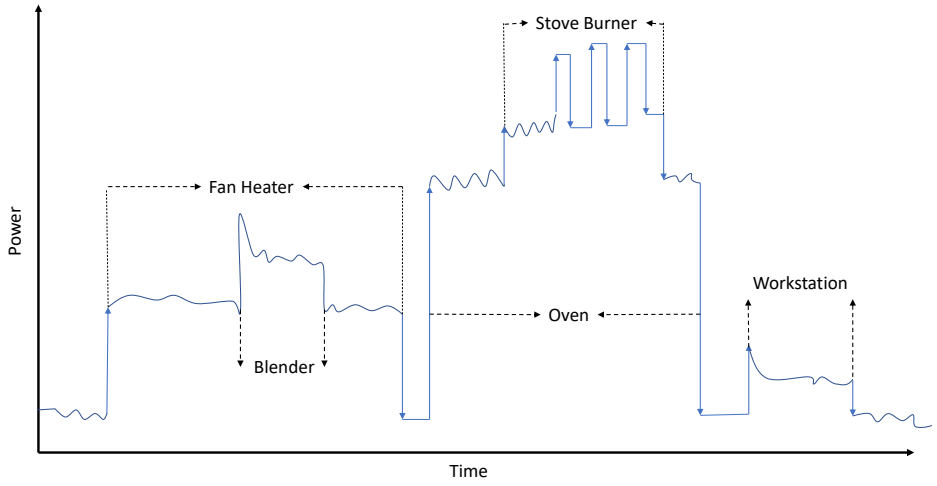

Figure 3.2: Example load curve showing multiple events of different appliances [48].

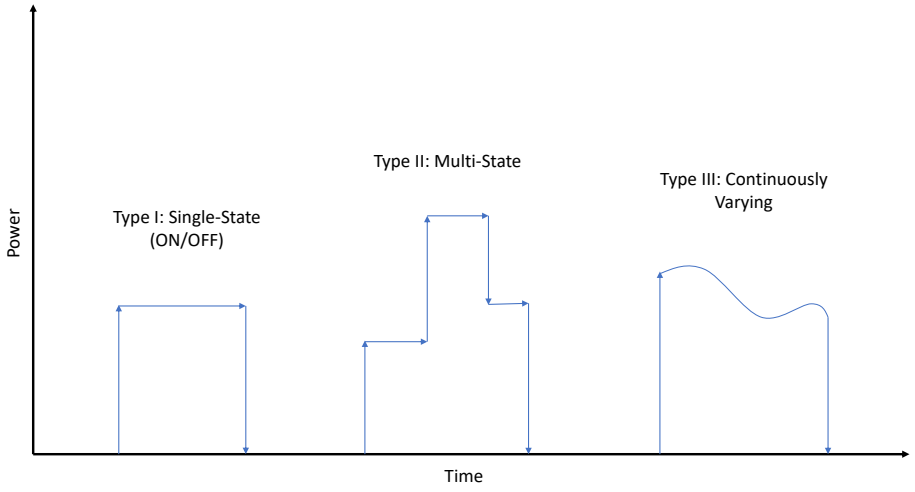

Figure 3.3: Different event types represented as approximated graphs [48].

3.3 Feature Extraction

In machine learning and pattern recognition, features are used to specify different classes. They should be chosen to define the different classes as precisely as possible. The features that are used with NILM algorithms can be categorized into steady-state and transient-state methods. Lately, more non-traditional features were included to support the traditional features and increase the disaggregation accuracy.

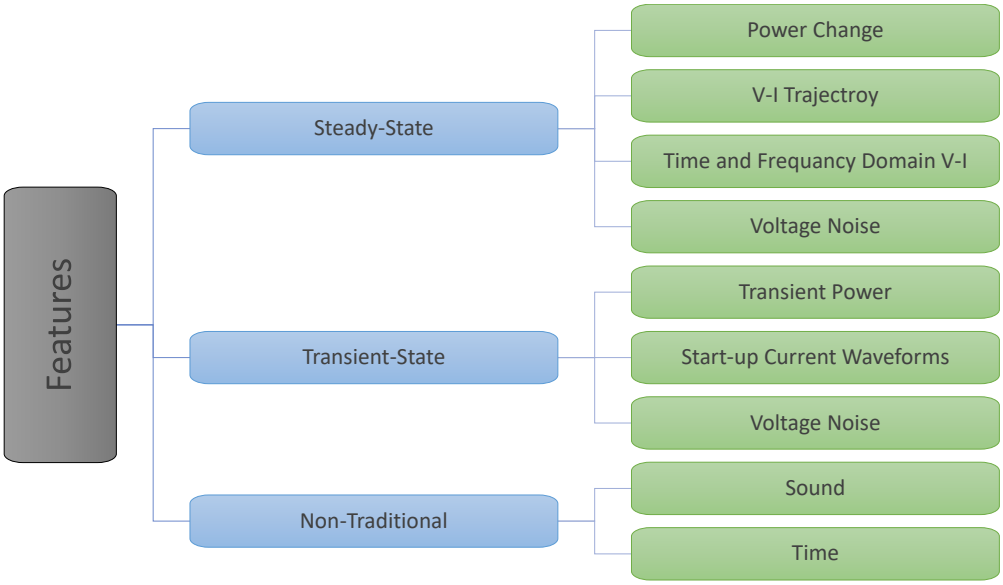

Figure 3.4: NILM feature categorization [48].

3.3.1 Steady-State Methods

Steady-state and transient-state methods are used to define the behaviour of systems or processes. A steady state means a time period in which the variables that describe the current process do not change. In NILM, a steady state occurs when an appliance has a steady power draw after its start-up or transition phase. The most commonly used steady-state features are real and reactive power. These features only require a low sampling rate and have a decent accuracy in detecting high powered appliances [36, 44, 34]. Therefore, they are mostly used to detect Type-I devices and have poor performance on low-powered appliances and Type-II, Type-III and Type IV loads. The time and frequency domain characteristics of V-I waveforms, like the root mean square of voltage and current, can be used to classify resistive, inductive and electronic loads. These features mostly require high sampling rates. V-I trajectories are also used to

generate shape-dependent appliance features like symmetry/asymmetry and looping directions. Usually, they are computationally expensive and can not be used in scenarios with overlapping loads of different appliances. The steady voltage noise can be used to classify motor-based appliances that generate a continuous noise in their voltage trajectory. As seen, every feature type has its advantages but also comes with some major drawbacks [48].

3.3.2 Transient-State Methods

A transient state occurs during the change between two steady states of a process or system. During this period, the system variables change to approach the next steady state. In NILM, a transient state could be either the start-up event of an appliance or the transition time of multi-state appliances. Transient events are harder to detect and require a high sampling rate. Therefore, transient features are computationally more expensive than steady-state features. Their advantage is that they usually offer a less overlapping feature space for each appliance class in comparison to steady-state features [9]. The first feature domain, the transient power profile, can be used to detect Type I, II and III loads. Start-up current peaks can be used to identify Type I and II appliances, even in multi-load scenarios. During transients, the voltage noise can also be used to detect devices equipped with SMPSs (Switch Mode Power Supply). Voltage noise can be affected by wiring architecture and is only suitable for appliances equipped with a SMPS [48].

3.3.3 Non-Traditional Methods

The current problem with NILM features is that they are not suitable to detect all four appliance types at once, as described in section 3.3.1 and 3.3.2. To compromise this, scientists try to combine different transient-state and steady-state features to improve the disaggregation accuracy [42]. Zeifman and Kim et al. utilize usage patterns of consumers for disaggregation. Their idea is to expand the NILM feature space with the on/off duration of appliances or simply the time of the day. Furthermore, the correlations between appliances that are often used in a specific order can be used as features to support NILM algorithms [45, 24].

3.4 Appliance Identification

There are two types of appliance identification algorithms: the ones that use supervised learning and the ones that use unsupervised or semi-supervised learning. Supervised learning requires labeled training datasets, leading to more costs and human

effort. Hence, unsupervised learning methods gained more interest to overcome these problems.

3.4.1 Supervised Learning

Supervised learning algorithms can be distinguished in optimization and pattern recognition methods. The optimization methods extract features from the unknown loads and compare them with the features of already known devices from a database. The algorithm tries to minimize the error between the features of the database and the load curves to find the best possible match. One major drawback are overlapping loads from multiple appliances at a time. If this occurs, multiple combinations of appliances have to be taken into account which increases the computation cost enormous. Moreover, loads from unknown devices, that are not stored in the database, can not be identified [48]. The pattern recognition algorithms are using multiple appliance specific features that are stored in a database and define the algorithm's parameters and structure.

3.4.2 Unsupervised Learning

As already mentioned above, unsupervised learning techniques do not require labeled training data and are ,therefore, more cost efficient and more attractive to the broad audience of NILM techniques. There are various approaches, mainly based on variations of Hidden Markov Models [16, 26, 24]. They all share the same drawback. They can not detect low-powered appliances during high-powered load waves. Also multi-state appliances with changing load signatures and different appliances with similar load waves bring these algorithms to their limit [48].

3.5 Challenges

To address the problems mentioned in section 3.4, researchers try to combine the steady-state and transient-state features with environmental information, such as the position of the consumer, sound waves recorded by microphones or lightning information using light sensors. Features, like the time of the day and the usage duration, are also combined with the NILM algorithms. All these new features need to be recorded with sensors that have to be manually installed in the measuring area which again leads to increased costs.
In addition, the evaluation of NILM algorithms has to be standardized in order to compare different approaches. The widely spread recognition accuracy is no suitable comparison method due to its inconsistent definition. Further, reference datasets have

to be used to reduce the distortions made by the measuring hardware and to give a constant appliance base to use for NILM algorithms [48].

Currently, there is no suitable feature space for NILM algorithms to detect all types of appliances mentioned in section 3.2. Every algorithm is focused to detect few appliance types or just one. The problems with low-powered and multi-state appliances, mentioned in section 3.4, could be addressed by using environmental features. Over the past decades, multiple datasets [27, 10, 2, 20, 25, 23] have been released for NILM testing purposes and have been accepted by the NILM community. In addition, the F_1-Score and Confusion Matrices have been used as a basic benchmark for performance and accuracy of developed algorithms [1, 15].

4 Dataset & Preprocessing

4.1 Building-Level fUlly-labeled Dataset (BLUED)

The dataset used in this paper for evaluating the algorithms performance, is the BLUED or Building-Level fUlly-labeled Dataset. This set has a sampling rate of 12kHz and was especially designed for event-based NILM, as explained in section 3.2. Other examples for NILM datasets are the non-event based REDD [25] or the BLOND [27], already mentioned in section 3.1. More event-based high frequency datasets are the WHITED [20] and the PLAID [10]. BLUED comes with one week of voltage and current data of a single family household. It was recorded during October 2011 in Pittsburgh, Pennsylvania in the United States. Residential buildings in the United States are usually supplied with a 3-wire, single-phase 240V/120V power distribution. The 240V primary single phase is used to create two 60Hz, 120V phases and also a neutral. The two 60Hz phases are further referenced as phase A and B and have a phase difference of 180 degree. Every transient change in phase A and B was documented and timestamped. The transient changes are considered as events and labeled with the corresponding appliance. The dataset consists of the two phases A and B, both with different amounts of appliances and events [2]. A table, showing the appliance name to index mapping, the average power consumption in watts, the number of appliance events and the phase, can be seen in table 4.1. In total, there are 2482 events to be used for testing, 904 in phase A and 1578 in B. Appliances that could not be metered with plug in meters or environmental sensors were measured in sub-circuits with index numbers over 200. Examples for this kind of appliances are hardwired or two-phase devices [2]. Unknown devices that were detected are labeled with index numbers over 1000. These appliances are not taken into account for testing and so a total of 2333 events were used for disaggregation. The index numbers in table 4.1 are further used in the algorithm evaluations of this paper.

4.2 Preprocessing

The two algorithms of Chapter 5 and 6 are based on using event information for disaggregation. Therefore, most of the voltage and current measurements of the 340

Table 4.1: BLUED appliance events and power consumption.

Index	Name	Average Power(W)	Events	Phase
101	Desktop Lamp	30	26	B
102	Tall Desk Lamp	30	25	B
103	Garage Door	530	24	B
108	Kitchen Aid Chopper	1500	16	A
111	Fridge	120	616	A
112	A/V Living room	45	8	B
118	Computer A	60	45	B
120	Laptop A	40	14	B
123	Basement Receiver/DVR/Blueray	55	34	B
127	Air Compressor	1130	20	A
128	LCD Monitor A	35	77	B
129	TV Basement	190	54	B
131	Printer	930	150	B
132	Hair Dryer	1600	8	A
134	Iron	1400	40	B
135	Empty living room socket	60	2	B
140	Monitor B	40	150	B
147	Backyard lights	60	16	A
148	Washroom light	110	6	A
149	Office Lights	30	54	B
150	Closet lights	20	22	B
151	Upstairs hallway light	25	17	B
152	Hallways Stairs lights	110	58	B
153	Kitchen Hallway light	15	6	B
155	Kitchen overhead light	65	56	B
156	Bathroom upstairs lights	65	98	A
157	Dining room overhead light	65	32	B
158	Bedroom Lights	190	19	A
159	Basement Light	35	39	B
204	Circuit 4	-	46	B
207	Circuit 7	-	38	A
209	Circuit 9	-	46	B
210	Circuit 10	-	99	B
211	Circuit 11	-	394	B
>1000	Unknown appliance	-	149	A+B

GB dataset can be ignored. BLUED provides lists of all events with their timestamps within the set. These event information were used to extract data windows around each event. These windows are then used to create individual feature spaces and test the two disaggregation algorithms. For the purpose of this paper, event windows of 0.5, 1.0 and 1.5 seconds were taken into account. The sinusoidal 60Hz signal sampled at 12kHz results in 200 data points per cycle. This leads to a total of 90 cycles in the 1.5 second event windows with 18000 data points and equivalent 30 cycles for the 0.5 second windows and 60 cycles for the 1.0 second windows. The 0.5 and 1.0 second windows have the triggering event located in the middle of the window. The 1.5 second window has 30 cycles or 0.5 second before and 60 cycles or 1.0 second after the event. Because most of the NILM features are evaluated per cycle, the event windows should start with the beginning of a new cycle. Therefore, every window is shifted so far to the right that the new window starts with a fresh cycle to ensure a valid feature computation. Figure 4.1 shows the different event window sizes for a refrigerator event extracted from the dataset. The vertical red line marks the start of the event. Clearly, there is a loss of information when shrinking the window. Later, it will be evaluated in how much this affects disaggregation accuracy. The PCA algorithm also needs one cycle before the event window to normalize the other cycles. Therefore, a single cycle is added to the beginning of each event window during the event extraction.

4.3 Data Augmentation

BLUED only has a small number of events for some appliance classes. The reason behind this is that collecting sub-metered information of appliances requires special equipment and high effort, as stated in 4.1. To compromise this, data augmentation was used. Data augmentation can help the classifier to get more variance of a class when only few training samples are available. In this paper, two augmentation techniques, presented by Jorde et al. [18], were used to enlarge the dataset.

1. Phase Shift (left): shifts the event window by one cycle to the left
2. Half-Phase Flip (left): shifts the signal by half a cycle to the left and changes the signs of the signal

A phase shift of a sample increases the time in variance of the classifier. The Half-Phase Flip increases the performance in detecting different variations of signals of the same class. These two methods are applied on every sample of the dataset during the training process of the models. Figure 4.2 shows the augmentation methods applied on the previous, in figure 4.1 presented event of a refrigerator.

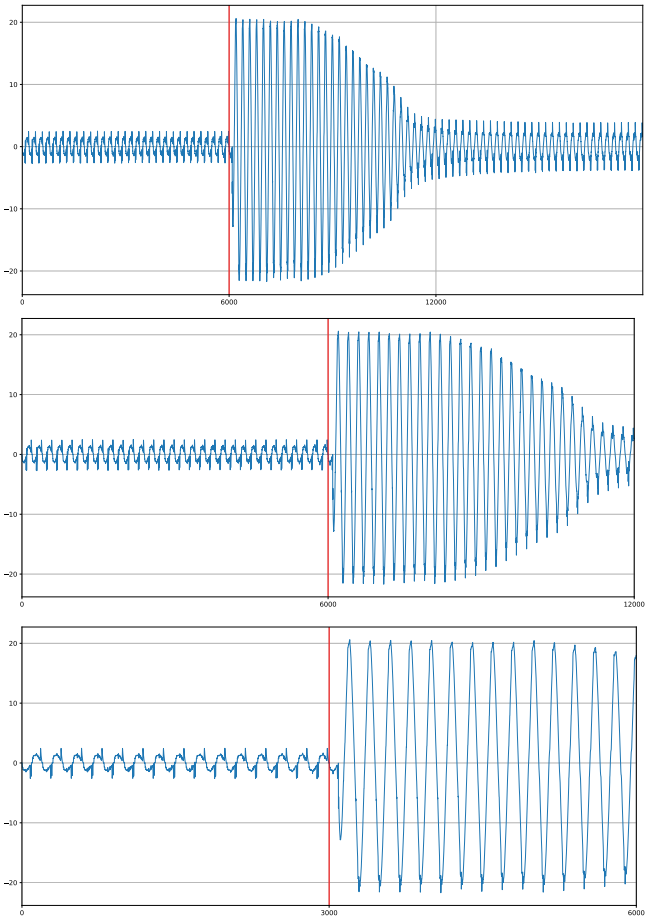

Figure 4.1: Different window lengths showing a refrigerator event.

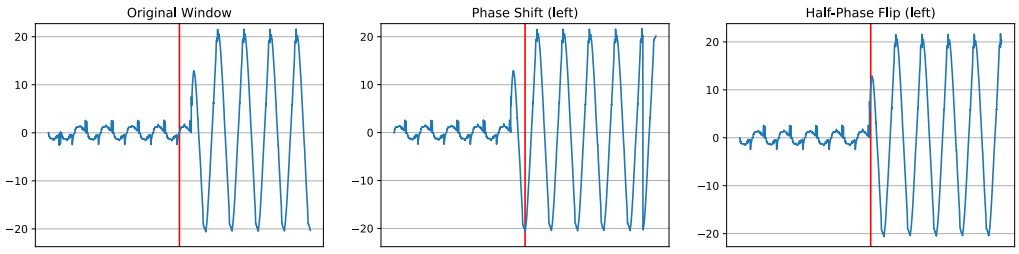

Figure 4.2: Augmentation methods applied on an event window of a refrigerator [18].

5 Current Peak Based Device Classification

The algorithm, presented in this chapter, is based on combining the peaks of current waves and was introduced by Jain et al. in 2017 [15]. It uses the Device Current Signature (DCS) as feature for classification. The authors of [15] tested multiple classifiers along with the DCS. The best performing classifier, Extra Trees, is further used for the classification in this paper.

5.1 Preprocessing

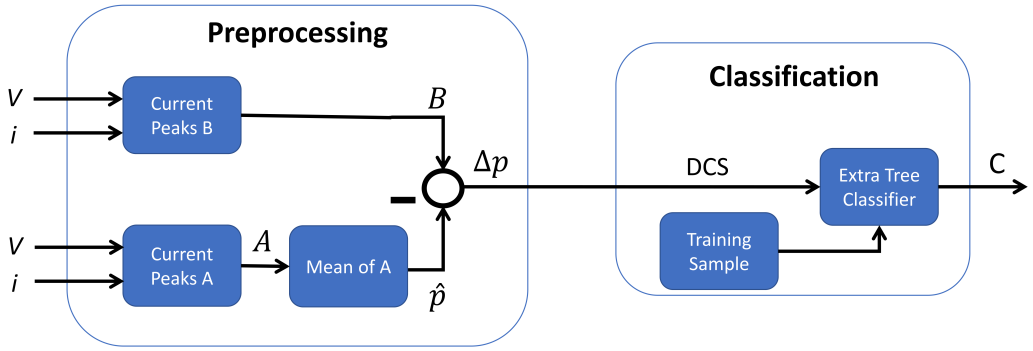

Figure 5.1: Representation of the preprocessing and classification process as block diagram.

As mentioned, the algorithm uses current peaks as features for classification. A current peak is the largest occurring current in one cycle of the device's frequency. BLUED has a 60 Hz signal sampled at 12 kHz which leads to a cycle length of 200 data points. Let y be a data point in a given cycle C, then the current peak p can be expressed as:

$$p = x : x \geq y \quad \forall y \in C \tag{1}$$

Jain et al. consider the 90 cycle event window as ideal [15]. Nevertheless, the 60 and 30 cycle versions are also tested in this paper. Therefore, let A be the collection of cycles

before the event ϵ occurs and B be the collection of cycles after the event. This leads to the cycle distribution shown in table 5.1.

Table 5.1: Cycle distribution for DCS

Cycles	A	B
30	15	15
60	30	30
90	30	60

With this cycle distribution, the set of all current peaks in A of any event window can be represented as:

$$A = p \in C_i \quad \forall i \in [1, n] \tag{2}$$

Where n is the amount of cycles in A. The set of B is similarly defined as A. This way A and B contain a single current peak for each cycle shown in table 5.1. BLUED only provides the aggregated current. Thus, the current peaks extracted in (2) contain current values of other appliances and not only the delta current that is obligatory for the appliance that caused the event. In order to obtain the relative magnitude of the current peaks of set B, the mean is taken from set A and later subtracted of every peak in B. So, the first step is to get the mean $\hat{p}$ of the steady-states of set A, as expressed in:

$$\hat{p} = \frac{1}{N} \sum_{i=1}^{N} p_i \quad \forall p_i \in A \tag{3}$$

Where N is the cycle length of the different window sizes in table 5.1. After the extraction of the steady-state means the Device Current Signature can be defined as:

$$\Delta p = p_i - \hat{p} \quad \forall p_i \in B \tag{4}$$

The DCS is now considered for training the Extra Trees classifier to test the performance of DCS as feature for energy disaggregation.

In figure 5.2 and 5.3, the DCS of the refrigerator event, introduced in Chapter 4, can be seen. Clearly, there is a loss of information going from 90 cycles to 30 cycles. In section 5.3, it will be evaluated in how much his affects disaggregation accuracy.

5.2 Classification

For classification, a basic Extra Tree classifier is applied on the data. The algorithm was introduced by Geurts et al. in 2006 [11]. It is based on randomizing Decision Trees [41]

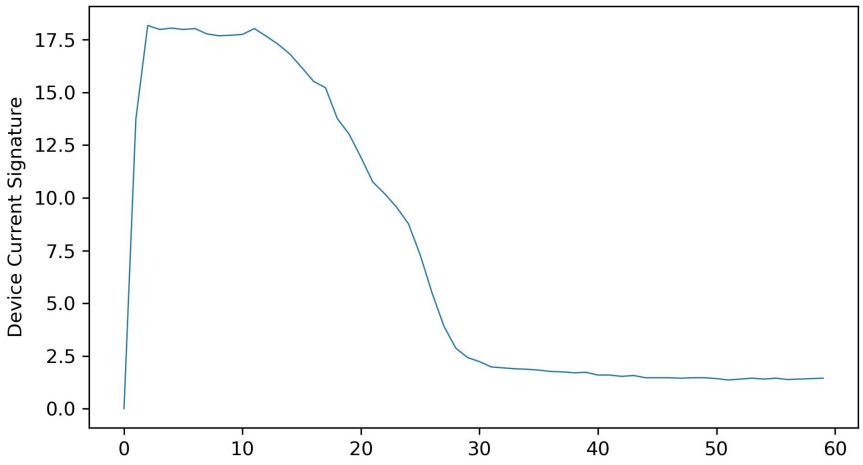

Figure 5.2: DCS of a refrigerator for the 90 cycle event window.

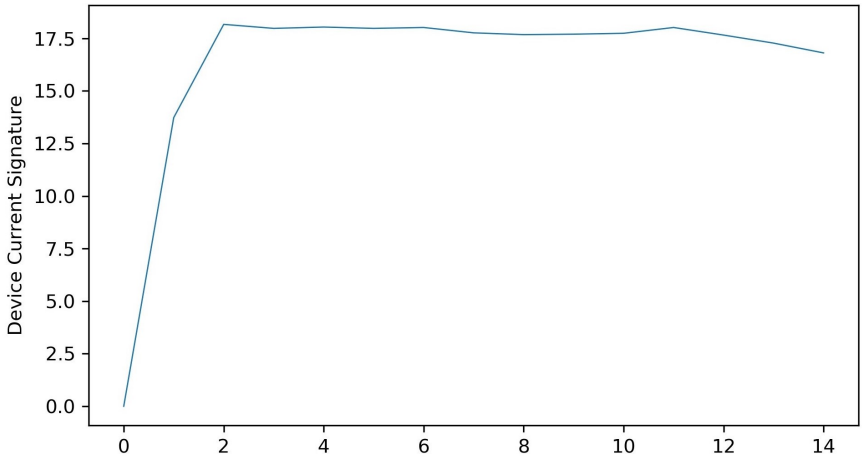

Figure 5.3: DCS of a refrigerator for the 30 cycle event window.

and can be used to solve regression and classification problems. The randomization occurs when a tree node is split up. There, the choice of attributes and cut-points are randomized. This way, the algorithm can build extremely randomized trees by ignoring the output values of the learning sample. The advantages of the algorithm are its performance and good accuracy without further tuning of parameters. More information can be found in [11, 39] and in [5] for the related Randomized Forests.

5.3 Experimental Results

Jain et al. reached an overall F_1-Score of 0.9315 for phase A and a score of 0.7889 for phase B [15]. These scores are used as benchmark for testing the implementation of this paper and the later introduced improvements. The algorithm was tested for all three window sizes of 30, 60 and 90 cycles. Because the BLUED dataset has some appliances with a small amount of events, the augmentation methods, presented in Chapter 4, were also applied on the data. In addition, different parameter variations were tested with the classifier. For separating the data into testing and training, a Stratified K-Fold algorithm was used. The two different split amounts 5 and 10 are considered. The amount of test and train samples can differ by the the amount of classes in the set for each run. This way, Stratified K-Fold can keep the percentage of each classes samples equal [39].
For evaluating the performance of the classifier, the F_1-Score is used as basic metric. It forms the harmonic mean of recall and precision and is evaluated as follows [39]:

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \tag{5}$$

The results of some different test runs are shown in the next table 5.2.

Window	Phase	Augment	Folds	Estimators	F1
30	A	True	10	10	0.914634
30	A	False	10	50	0.923077
60	A	False	10	50	0.923077
60	A	False	10	100	0.935897
90	A	False	10	500	0.935897
90	A	False	10	100	0.935897
30	A+B	False	10	50	0.747706
30	A+B	False	10	500	0.761062

Continued on next page

Window	Phase	Augment	Folds	Estimators	F1
60	A+B	False	10	50	0.765487
60	A+B	False	10	100	0.765487
90	A+B	False	10	500	0.769912
90	A+B	False	10	100	0.775229
30	B	False	10	500	0.736111
30	B	False	10	100	0.739437
60	B	True	10	500	0.757303
60	B	False	10	10	0.760563
90	B	True	10	500	0.764045
90	B	False	10	100	0.767296

Table 5.2: Extra Tree results only using DCS as feature.

As seen in the table 5.2, the window size does not have a huge impact on the performance with all variations performing equally well. Data augmentation did not have an impact either. The algorithm recreation of [15] performed equally well with a slightly better performance in phase A and a slightly worse performance in phase B.

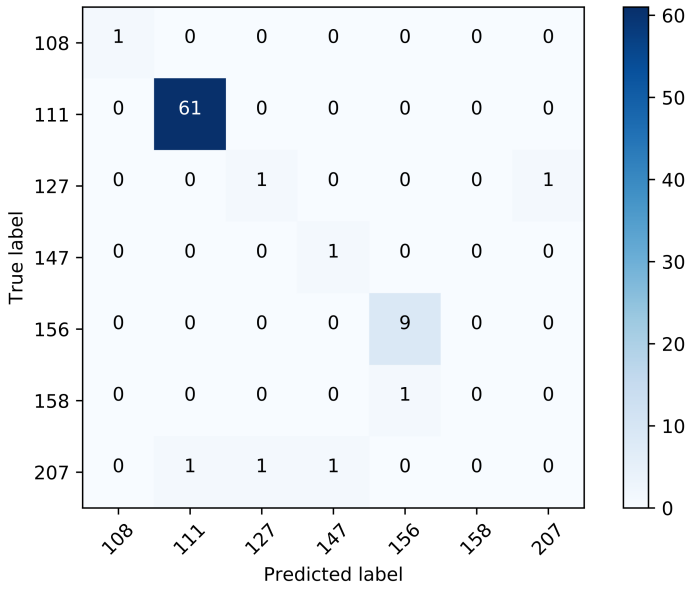

Figure 5.4: Confusion matrix of the best result in phase A with a 90 cycle window only using DCS.

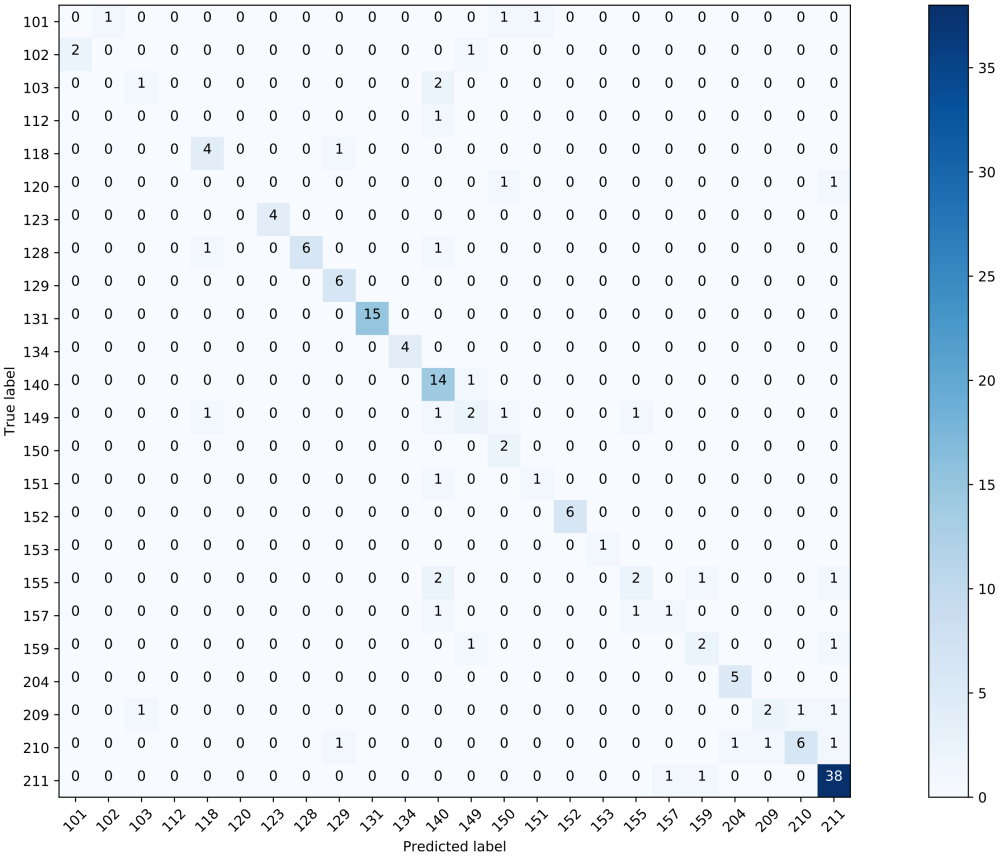

Figure 5.5: Confusion matrix of the best result in phase B with a 90 cycle window only using DCS.

The two confusion matrices 5.4 and 5.5 show the predicted and the true appliance labels of the best classification runs in phase A and B. The appliance names are represented by the index numbers introduced in table 4.1. The diagonal line of the matrix represents the correct classified appliances.

5.4 Extended Experimental Results

For further evaluation of the Extra Tree classifier and the Device Current Signature, a multi-dimensional feature space is created. Therefore, multiple features presented in [19] are used. This paper presents a NILM feature study on four datasets: PLAID [10], WHITED [20], BLUED [2] and UK-DALE [23]. The partly self developed and already established features were tested with four different classifier types: Linear Discriminant Analysis, K Nearest Neighbours, Support Vector Machines and Boosted Decision Trees. The two features, Admittance Over Time (AOT) and Current Over Time (COT), are the best performing single features on the BLUED dataset with a F_1-Score of 0.6. These two features are especially good for detecting appliances that do not have a steady, but rather a decreasing power consumption after their initial start-up. The COT is built by combining the Root Mean Squared current of each cycle of the event window into a multi-dimensional vector. Dividing each cycle of the COT by the associated voltage creates the more to voltage fluctuations resistant feature vector AOT. Let n be the event window length and I_i, U_i the RMS of one cycle, then AOT and COT can be computed as follows:

$$COT = [I_1, I_2, ..., I_n] \tag{6}$$

$$AOT = [\frac{I_1}{U_1}, \frac{I_2}{U_2}, ..., \frac{I_n}{U_n}] \tag{7}$$

The two graphs 5.6 and 5.7 show the AOT and COT for the refrigerator event that was also used for displaying the DCS in 5.2 and 5.3. Definitely, there are similarities between the COT/AOT and DCS. The difference is that DCS uses the current peaks and COT/AOT uses the RMS of the current cycles.

In addition to COT and AOT, the already commonly used Apparent Power S, Active Power P and Reactive Power Q are used for further evaluation. Their computation formulas are shown in the following equations. The Phase Shift ϕ is the phase angle difference between current and voltage.

$$I_{rms} = \sqrt{\frac{1}{T}\int_0^T i(t)^2 dt} \tag{8}$$

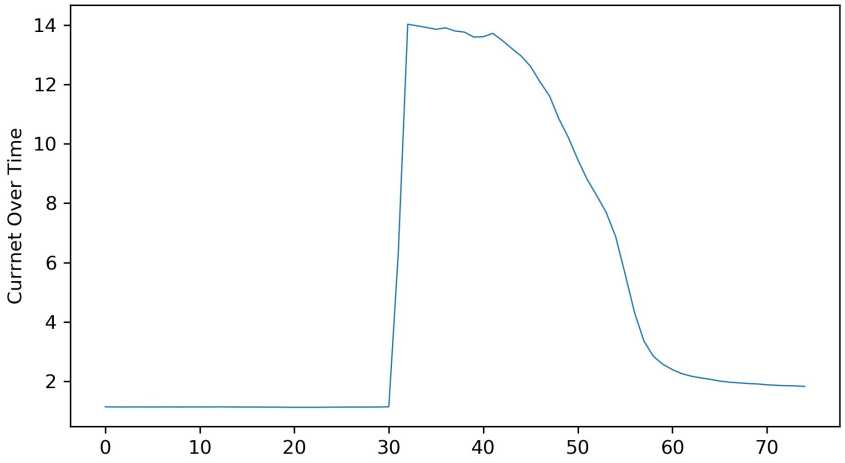

Figure 5.6: COT of a refrigerator event.

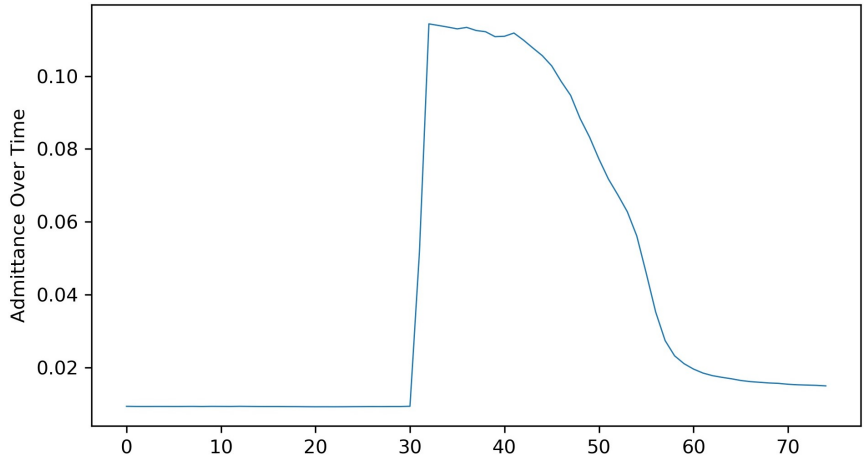

Figure 5.7: AOT of a refrigerator event.

$$S = V_{rms} * I_{rms} \tag{9}$$

$$P = V_{rms} * I_{rms} * \cos(\phi) \tag{10}$$

$$Q = V_{rms} * I_{rms} * \sin(\phi) \tag{11}$$

In this paper, the Phase Shift ϕ is not calculated due to complexity and time reasons. Therefore, the approximations (12) for Active Power P and (13) for Reactive Power Q are used. N is the length of the event window.

$$P = \sum_{n=0}^{N} (i[n] * v[n])/N \tag{12}$$

$$Q = \sqrt{S^2 + P^2} \tag{13}$$

Equivalent to the evaluation shown in 5.2, the algorithm was tested with different multi-dimensional feature combinations of the features presented above.

Window	Phase	Features	Augment	Folds	Estim.	F1
30	A	DCS, AC	False	10	500	0.936709
30	A	COT, AOT, AC	True	10	10	0.939024
60	A	DCS, COT, AOT, AC	False	10	10	0.961538
60	A	DCS, AC	False	10	50	0.961538
90	A	DCS, AC	False	10	10	0.948718
90	A	DCS, COT, AOT, AC	False	10	50	0.948718
30	A+B	DCS, AC	True	10	100	0.829233
30	A+B	DCS, COT, AOT, AC	False	10	50	0.846154
60	A+B	DCS, COT, AOT, AC	False	10	100	0.828054
60	A+B	DCS, AC	False	10	100	0.832579
90	A+B	DCS, COT, AOT, AC	True	10	500	0.831429
90	A+B	DCS, AC	False	10	50	0.840708
30	B	DCS, AC	False	10	500	0.774648
30	B	DCS, COT, AOT, AC	False	10	500	0.774648
60	B	DCS, AC	True	10	500	0.764835
60	B	DCS, COT, AOT, AC	True	10	500	0.767033
90	B	DCS, AC	True	10	500	0.786813
90	B	DCS, COT, AOT, AC	False	10	10	0.791667

Table 5.3: Extra Tree using a multi-dimensional feature space.

With the help of the additional features presented above, the overall disaggregation performance could be increased across all phases. Especially in phase A, a F_1-Score of 96% was achieved ,which is an increase of 3% to the previous results. In phase B, the performance also increased by nearly 3%. The combined results of A and B also increased by 7%.

5.5 Conclusion

The Extra Tree classifier is an excellent choice to combine with the Device Current Signature. Using the current peak information of all cycles in the event windows, respectively to the F_1-Score, allows the classifier to create unique appliance specific models with nearly no overlapping among each class. With the Extra Tree classifier, an offline model can be trained, allowing the classifier to be directly exported into new households without further training. The already good results of the classifier, with a F_1-Score of 93% in phase A and 76% in phase B, could be further increased using Current Over Time, Admittance Over Time and the SPQ trajectories along with the Device Current Signature. The other parameters, like the amount of estimators used by the tree, did not influence the accuracy significantly. Augmentation of the dataset did sometimes lead to good results, but the majority of tests tended to produce better results without augmentation.

6 (S)PQD-PCA Device Classification

The PQD-PCA classification presented in [1] is based on the reduction of the feature space using Principle Component Analysis (PCA) and evaluating vector norms to determine a samples appliance class.
For the purpose of this paper, only the data preprocessing and classification of events was accomplished. This means the event detection part of [1] was skipped and handled as a precondition. This way, it is possible to get the best possible performance of the classifier without any falsely labeled events. Further PCA information can be found in [17].

6.1 Preprocessing

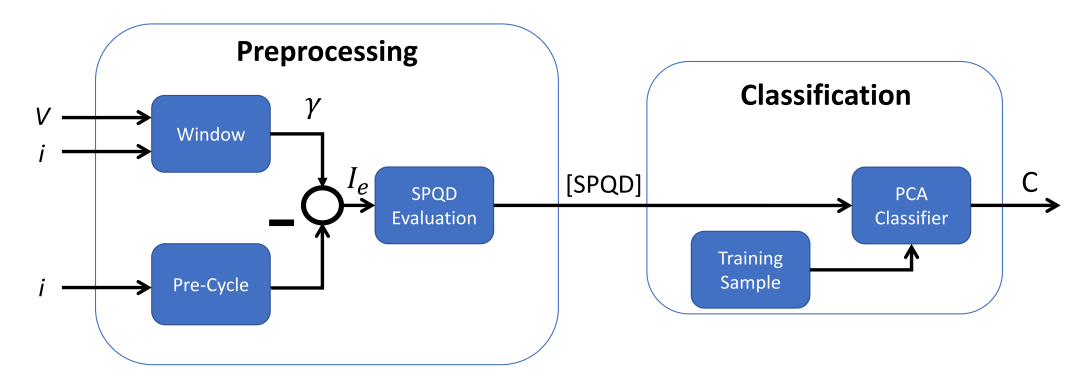

Figure 6.1: Representation of the preprocessing and classification process as block diagram [1].

The proposed algorithm uses Real, Reactive and Distortion power of the signal to classify appliances. In addition, the PQD curve trajectories are characterized during the transient state as well as the previous and the next steady states. With the combination of transient and steady states, it is attempted to achieve higher classification accuracy. The first part of the preprocessing process is to extract the delta values of current and voltage of each event window. This is necessary because BLUED only provides

the aggregated current and voltage, and not the isolated signal that is produced by a specific appliance causing the recorded event. Therefore, as already mentioned in chapter 4, one additional cycle before each event window is needed to create the approximate current wave that is only produced by the recorded appliance. Let i be a cycle and i_e the cycle in which the event e took place, then the cycle vector γ with length w can be described as shown in equation (14). As seen in figure 6.1, the steady cycle before an event window γ is subtracted of all other aggregated cycles of the event window. This way, only the delta current wave $\mathbf{I}_e$ remains (15). Note that this is only necessary for the current wave because the voltage wave should not be affected by the amount of appliances connected to the phase.

$$\gamma = \{\mathbf{i}_{e-\frac{w}{2}}, ..., \mathbf{i}_e, ..., \mathbf{i}_{e+\frac{w}{2}}\} \tag{14}$$

$$\mathbf{I}_e = \gamma - \{\mathbf{i}_{e-\frac{w}{2}-1}, ..., \mathbf{i}_{e+\frac{w}{2}-1}\} \tag{15}$$

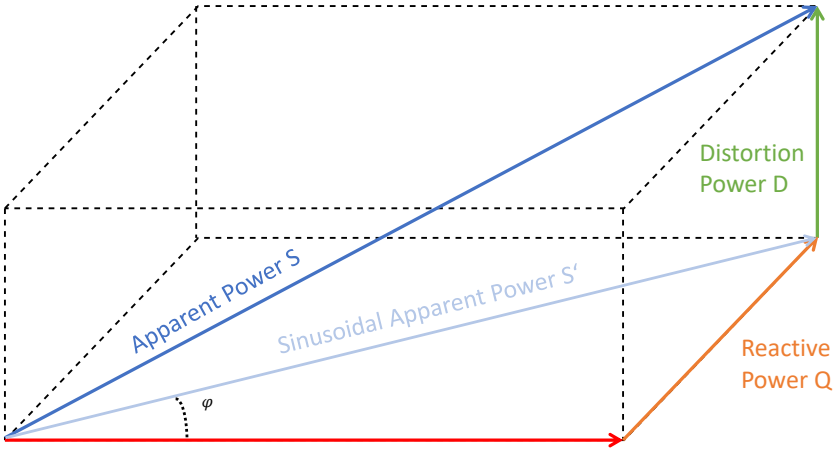

Figure 6.2: 3-dimensional representation of the SPQD load behaviour [1].

The authors of [1] propose a window length of 60 cycles as ideal. Nevertheless, as mentioned in section 4, the algorithm is also tested with window length of 90 and 30 cycles. The delta current extraction remains equivalent to the one with 60 cycles proposed by Alcalá et al. [1], except that the event cycle i_e is situated at one third of the event window for the 90 cycle version. After every event window is transformed as expressed in equation (15), the actual calculation of the SPQD trajectories is performed. Every SPQD evaluation represents a point in the 3-dimensional power cube 6.2. At first, the root mean square of voltage and current is computed, where T is the fundamental

period that corresponds to the utility frequency (16). Then Apparent S, Active P, Reactive Q and Distortion Power D are calculated with the equations (17, 18, 19, 20).

$$I_{rms} = \sqrt{\frac{1}{T}\int_0^T i(t)^2 dt} \tag{16}$$

$$S^2 = P^2 + Q^2 + D^2 \qquad D = \sqrt{S^2 - P^2 - Q^2} \tag{17}$$

$$S = V_{rms} * I_{rms} \tag{18}$$

$$P = V_{rms} * I_{rms} * \cos(\phi) \tag{19}$$

$$Q = V_{rms} * I_{rms} * \sin(\phi) \tag{20}$$

Because the phase shift ϕ is not trivial to compute, the approximations of equation (22) and (21) for Reactive Power Q and Active Power P are used. N represents the total amount of samples in the cycle. To avoid abrupt changes in the PQD trajectories, the evaluation of PQD is done every four cycles of the utility frequency with an overlapping of 3 cycles. The authors of [1] also propose to use equation (23) to avoid voltage drops for the PQD trajectories. Here, W is the total amount of samples of the event window.

$$P = \sum_{n=0}^{N} (i[n] * v[n])/N \tag{21}$$

$$Q = \sqrt{S^2 + P^2} \tag{22}$$

$$P_i = -P_i * \frac{P_i^2}{S_i^2} \qquad \forall i = 0, 1, ..., W-1 \tag{23}$$

The now obtained collection of PQD trajectories per class contain significant appliance type information. The evaluated delta values for the current waves are able to create appliance specific trajectories during steady states and provide a non-overlapping feature space. The transient states during start-ups of appliances are strongly influenced by circuit layout and configuration. Therefore, transient features have a more overlapping feature space with other classes. The combination of both, transient and steady features, provide a reasonable basis for classification.

6.2 Classification

The now obtained set of PQD trajectories per class have to be further transformed to train the PCA model. So first a normal Principal Component Analysis is performed that is further explained. The computation of P, Q and D with the equations (17, 21,

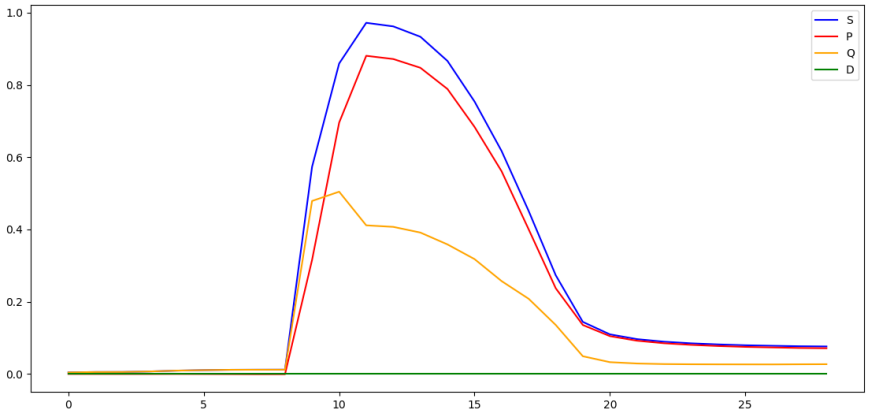

Figure 6.3: Scaled SPQD trajectories of the refrigerator event.

22) provides a three w-dimensional vector per event m. These three vectors can be rearranged as 3 by w wide matrix:

$$[\mathbf{PQD}]_m = [[P_0, ..., P_{w-1}]^T, [Q_0, ..., Q_{w-1}]^T, [D_0, ..., D_{w-1}]^T] \tag{24}$$

This PQD matrix is then rearranged as a $3w$ by 1 vector τ:

$$\tau_m = \{\mathbf{P}, \mathbf{Q}, \mathbf{D}\}^T \tag{25}$$

Now, there is a training set of M $3w$ by 1 vectors τ_m for each appliance class in the dataset. Then, for every class the averaged vector ψ_M can be computed with the following equation:

$$\psi_M = \frac{1}{M} \sum_{m=0}^{M-1} \tau_m \tag{26}$$

The average vector ψ_M also has a dimension of $3w$ by 1 and further ψ_M can be subtracted of every PQD vector τ_m per appliance class. This way only the variance ϕ_m remains.

$$\phi_m = \tau_m - \psi_M \qquad \forall m = 0, 1..., M-1 \tag{27}$$

Instead of the in PCA algorithm primarily used covariance matrix C, the authors of [1] propose the scatter matrix S.

$$C_T = \frac{1}{M-1} \sum_{m=0}^{M-1} \phi_m \phi_m^T \tag{28}$$

$$S_T = \sum_{m=0}^{M-1} \phi_m \phi_m^T \tag{29}$$

The two different matrix types (29) and (28) for the PCA algorithms were both tested with resulting in nearly no difference. To follow [1], the scatter matrix S was further used in the algorithm's evaluation. Now, the dimension reduction of the scatter matrix S_T can be carried out. Therefore, the $3w$ by $3w$ scatter matrix S_T is projected in a $3w$ by l dimension with $l < 3w$. PCA uses the l largest eigenvectors u of S_T to reduce the feature dimension. This way, the new scatter matrix $U^T S_T U$ has a maximal determinant. The eigenvectors u_1 to u_l are the so called principal components with u_1 providing the highest variance. The l largest eigenvectors u, chosen by their eigenvalues, form the new matrix U_{opt}:

$$U_{opt} = argmax_U[U^T S_T U] = [u_1, u_2..., u_l] \tag{30}$$

Yet, the classification and training process resulted in the matrix U_{opt} for every class. After this step, the training process of the PCA model is completed and both U_{opt} and ψ are stored in the model. Now, consider a new sample $\tau_e = [PQD]_e$ of the testing set E needs to be classified. The first step is to transform τ_e into the new feature space:

$$\Omega_e = U_{opt}^T \phi_e = U_{opt}^T (\tau_e - \psi) \tag{31}$$

Where ψ is the in (26) calculated mean vector stored in the PCA model per class. In the next step, the reconstruction matrix $\hat{\psi}$ can be computed with the following equation:

$$\hat{\psi}_e = \hat{\tau}_e - \phi = U_{opt} \Omega_e \tag{32}$$

The dimensional reduction, described in 30 with the u largest eigenvectors, results in a loss of information in the reconstruction matrix $\hat{\psi}_e$. This loss can be expressed with the vector norm in:

$$\epsilon_e = ||\psi_e - \hat{\psi}_e|| \tag{33}$$

The construction of U_{opt} in (30) uses the l largest eigenvectors. Thus, only a small amount of variance is lost. Therefore, the construction of (32) should barley lose any information. This leads to a minor loss expressed in (33). Otherwise, the appliance τ_e has a different variance distribution and belongs to a different class. A sample τ_e can be classified in a certain class when the error ϵ_e is smaller than a previously set threshold. With the help of a heuristic threshold, outliners could be found and skipped during the classification process. The authors of [1] propose to use the average error that occurs during the training per class as heuristic threshold multiplied by 5. Another method could be to find the smallest error ϵ_e across all appliance classes C in the dataset. This method increases the computation time, especially if there is a high number of classes in the model. In this paper, no heuristic threshold was used because the number of

classes in the BLUED is not exceptionally high. The sample τ_e is simply classified as the appliance type where the smallest error ϵ_e occurs during the construction of (32) across all classes C. This method is represented as follows:

$$c = min \quad \epsilon_{e,c} \qquad \forall c \in C \tag{34}$$

6.3 Experimental Results

Alcalá et al. tested the PQD-PCA with the PLAID [10] dataset and achieved an overall F_1-Score of 90,6%. They used 97% of the variance and only a 3% loss in creating U_{opt} [1]. In this paper, the algorithm's performance is tested on all events of the BLUED dataset as preprocessed in Chapter 4. This means there are three different event window sizes with 30, 60 and 90 cycles. In total, 2333 events are used for training and testing. For train and test splitting, a Stratified K-Fold algorithm with two different amount of splits was used. Throughout the testing, also different variance to loss ratios in the construction of U_{opt} (30) were tested. Further, tests with and without data augmentation were performed.
The first tests on the BLUED dataset with the PQD-PCA classifier did not seem very promising. Therefore, before going deeper into testing, the Apparent Power S was added to the feature list, making it a SPQD-PCA classifier. In addition, scaling was applied on the SPQD curves. PCA tries to maximize the feature variance along the feature axis. Therefore, it can be useful to scale the data when using PCA. The standard score τ_s of a sample τ is computed as follows [39]:

$$\tau_s = \frac{\tau - u}{s} \tag{35}$$

Where u is the mean of the training samples and s is the standard deviation of the training samples.
For evaluating the performance of the classifier, equally to Chapter 5, the F_1-Score is used as basic metric. It forms the harmonic mean of recall and precision and is evaluated as follows [39]:

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \tag{36}$$

The test results of the SPQD classifier with different parameter variations can be seen in table 6.1. The algorithm was tested for phase A and B and both phases combined. The data augmentation presented in Chapter 4 did not bring any advantages over the non-augmented dataset. Rather, augmenting decreased the overall disaggregation accuracy.

Window	Phase	Augment	Folds	Variance	F1
30	A	False	10	0.85	0.756410
30	A	False	10	0.99	0.794872
60	A	False	5	0.85	0.721212
60	A	False	5	0.90	0.739394
90	A	False	5	0.97	0.769697
90	A	False	10	0.99	0.784810
30	A+B	False	10	0.90	0.315789
30	A+B	False	10	0.85	0.346491
60	A+B	False	10	0.90	0.344037
60	A+B	False	10	0.85	0.362385
90	A+B	False	5	0.99	0.538627
90	A+B	False	10	0.99	0.539474
30	B	False	10	0.85	0.354167
30	B	False	10	0.97	0.368056
60	B	False	5	0.97	0.445183
60	B	False	10	0.97	0.464789
90	B	False	10	0.99	0.527397
90	B	False	10	0.97	0.527397

Table 6.1: SPQD-PCA results only using SPQD trajectories as features.

Across phase A, the length of the event windows does not play a significant role. All three windows performed similarly well with a F_1-Score of up to nearly 80%. For phase B, the 30 cycle windows did not work at all, providing only a score of 36%. The 90 cycle windows performed best for phase B with up to 52%. Especially in phase B, it seems that some devices provide significant appliance information even after 0.5 seconds of the event occurrence. The performance loss of the SPQD classifier in comparison to [1] can probably be explained with the usage of a different dataset and the approximations of S, P, Q and D, as explained with equations (13)(12). The approximation leads to the Distortion Power D being zero for most of the event window. Hence, the Distortion Power does not offer any information for the disaggregation algorithm, and could be skipped in this implementation to decrease computation time.

The two confusion matrices 6.4 and 6.5 show the predicted and the true appliance labels of the best classification runs in phase A and B. The appliance names are represented by the index numbers introduced in table 4.1. The diagonal line of the matrix represents the correct classified appliances.

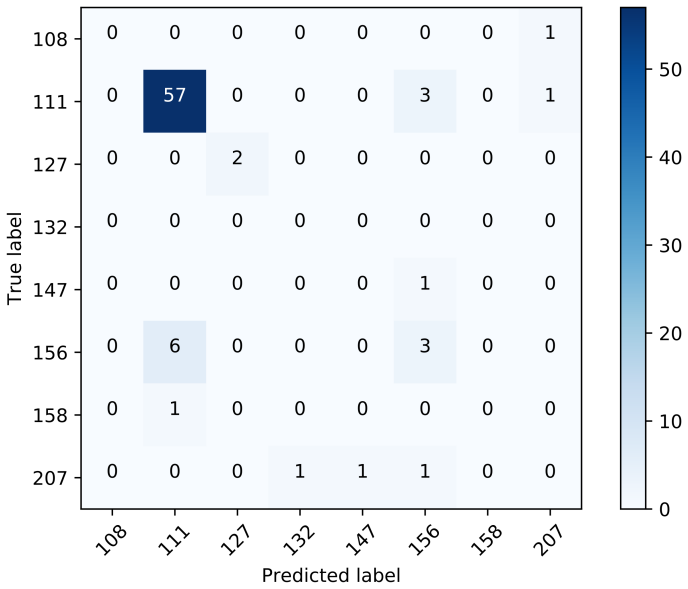

Figure 6.4: Confusion matrix of the best result in phase A with a 30 cycle window.

6.4 Extended Experimental Results

Equivalent to the extended evaluation of the Extra Tree classifier, the SPQD-PCA is tested with the features presented in Chapter 5. Current Over Time (COT), Admittance Over Time (AOT) and Device Current Signature (DCS) were used in combination with the AC-Powers SQPD to increase the algorithm's performance. The best results for all phases and window lengths can be seen in table 6.2.

Window	Phase	Features	Augment	Folds	Var.	F1
30	A	AC, DCS, COT, AOT	False	10	0.99	0.500000
30	A	AC, DCS	False	10	0.99	0.616279
60	A	AC, DCS, COT, AOT	False	10	0.99	0.548780
60	A	AC, DCS	False	5	0.99	0.709091
90	A	AC, DCS, COT, AOT	True	10	0.99	0.652000
90	A	AC, DCS	False	10	0.99	0.719512
30	A+B	AC, DCS	False	10	0.97	0.278481
30	A+B	AC, DCS, COT, AOT	False	10	0.99	0.298387
60	A+B	AC, DCS, COT, AOT	True	10	0.97	0.289875

Continued on next page

Window	Phase	Features	Augment	Folds	Var.	F1
60	A+B	AC, DCS	False	10	0.99	0.362385
90	A+B	AC, DCS, COT, AOT	False	10	0.97	0.402655
90	A+B	AC, DCS	False	10	0.99	0.486726
30	B	AC, DCS, COT, AOT	False	10	0.99	0.401408
30	B	AC, DCS	False	10	0.99	0.417808
60	B	AC, DCS, COT, AOT	True	10	0.99	0.453961
60	B	AC, DCS, COT, AOT	False	10	0.97	0.478873
90	B	AC, DCS, COT, AOT	False	10	0.99	0.471831
90	B	AC, DCS	True	10	0.99	0.486667

Table 6.2: PCA results using a multi-dimensional feature space.

As seen in the table, a multi-dimensional feature space of different feature combinations does not help the PCA classifier at all. The classification results of all window to phase combinations are equal or worse. The largest loss occurs in phase A with a total of 7%.

6.5 Conclusion

The SPQD-Classifier provides a valid opportunity for the disaggregation of electrical loads. With a F_1-Score of nearly 80% on the BLUED set, the algorithm does not perform as well as the Extra Tree classifier. Still, the SPQD trajectories offer a good feature space with only a small amount of overlapping among all appliance classes. The approximations of SPQD in equations (21) and (22) are decent, but leading to the loss of Distortion Power D during the evaluation. The usage of Principal Component Analysis offers a good scalability for lager feature spaces because of its variable compression rate. Adding Current Over Time, Admittance Over Time and the Device Current Signature did not result in an increase of performance, but rather decreased the overall results in both phases about 3%. An other advantage of the PCA model is its ability to use offline training. In addition, the model can be extended by new appliances at any time without the need of retraining the whole model. Augmentation also does not seem to help the classifier, equivalent to the previous algorithm. Augmented runs always significantly under-performed the runs without augmented data. Changing the variance of the PCA for dimension reduction had no severe impact on accuracy either. A variance of 97% seems to be a good trade-off in disaggregation performance and accuracy.

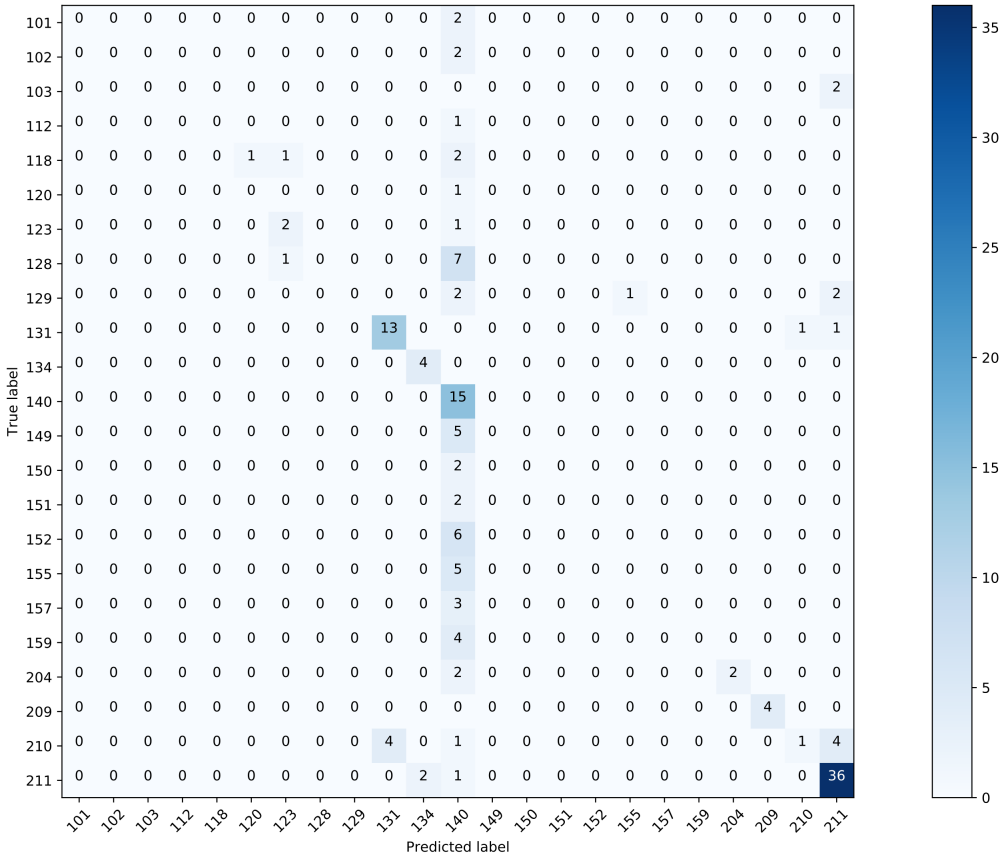

Figure 6.5: Confusion matrix of the best result in phase B with a 90 cycle window.

7 Future Work

As already mentioned in chapter 3, Non-Intrusive Load Monitoring is still a relatively new research area and, therefore, does not offer the state-of-the-art solution for all energy disaggregation problems. The Extra Tree classifier 5 and the SPQD-PCA 6 presented in this paper can still be improved and more testing can be done. As proven, the Extra Tree classifier's accuracy can be improved by using multiple features. Hence, more feature combinations should be tested and compared to the results of this paper. For the PCA algorithm, a multi-dimensional feature space could not improve the accuracy. Still, other feature combinations could be tested and other variance distributions for the dimension reduction. Especially the accuracy of appliances in phase B should be improved. In Chapter 6 and by Alcalá et al. [1], a method is presented to discard outliners during the training and classification process. This includes the usage of a heuristic threshold computed for each class during the training. This way, appliance events that significantly show other load patterns than normal can be discarded for both training and classifying. This could lead to an increase of the PCA's accuracy. Moreover, the classifiers should be tested on more scenarios of real world energy data to validate their performance, not primarily on the BLUED dataset. Only the PCA algorithm was tested on an other dataset by the authors of [1]. In this paper, the PCA showed great results on the PLAID dataset. These good results could not be strengthened with the performance on the BLUED dataset. This leads to the conclusion that there is a severe difference of classification difficulty among the energy datasets. Therefore, both algorithms should be tested on more data including PLAID [10], WHITED [20], BLOND [27] and UK-DALE [23], and their performance should be compared.

8 Comparison & Summary

This paper presents two event-based, supervised energy disaggregation algorithms using offline training. The Extra Tree classifier in Chapter 5 and the (S)PQD-PCA in Chapter 5 show two different approaches for disaggregating real world household energy data. Both algorithms produce valid classification results on the BLUED dataset measured in multiple runs with different parameters. In total, each algorithm was tested with 720 test runs with various parameter distributions. The algorithms were tested with three event window lengths of 30, 60 and 90 cycles and the six features Current Over Time, Admittance Over Time, Device Current Signature and Apparent, Real and Reactive Power. For the PCA, the Distortion Power was also considered as feature which did not hold any information due to approximations in the AC-Power evaluation (21)(22). With data augmentation, presented in Chapter 4, no increase of disaggregation accuracy was achieved in both algorithms. A Stratified K-Fold algorithm was applied to split the BLUED in test and training sets using 5 and 10 folds. For most of the tests a larger amount of folds resulted in higher accuracy. Along with the PCA, different variance ratios for dimension reduction were tested, 0.99, 0.97, 0.90 and 0.85. For the most part, a higher percentage achieved better results. 97% seems to be a good trade of in computation costs and accuracy. For the Extra Tree classifier, different amounts of estimators were tested, 10, 50, 100 and 500. More estimators normally resulted in higher accuracy with nearly no performance difference.

Using the standalone feature DCS, the Extra Tree classifier reached a F_1-Score of 93% in phase A and 78% in phase B. Using multiple features increased the accuracy to 96% and nearly 80%. The PCA performed best when only the scaled SPQD trajectories were used for classification. A F_1-Score of nearly 80% for phase A and 52% for phase B was reached. Adding more features always resulted in a lower accuracy. For both algorithms, the appliances in phase B were harder to classify than the appliances in phase A. This is due to a higher variety of appliances and view training samples for some classes in this phase. Especially that the PCA underperformed the Extra Tree classifier by 30% infers that the SPQD trajectories do not offer a proper non-overlapping appliance space.

All tests were performed on a 16 core server with 40GB of RAM. Loading and preprocessing the data took longest with up to 20-30 minutes depending on features and whether the data being augmented or not. For both algorithms, training and

testing was achieved in under 10 seconds. For the event extraction algorithm that was applied on the dataset to extract the different windows, the 40 GB of RAM often limited the performance. The extraction of all 2482 events took around half a day on the same server used for classification.

List of Figures

List of Tables

Bibliography

[1] J. M. Alcalá, J. Ureña, Á. Hernández, and D. Gualda. "Event-Based Energy Disaggregation Algorithm for Activity Monitoring From a Single-Point Sensor." In: *IEEE Transactions on Instrumentation and Measurement* 66 (2017), pp. 2615–2626.

[2] K. Anderson, A. Ocneanu, D. R. Carlson, A. Rowe, and M. Bergés. "BLUED : A Fully Labeled Public Dataset for Event-Based Non-Intrusive Load Monitoring Research." In: 2012.

[3] M. Bergés, E. Goldman, H. S. Matthews, and L. Soibelman. "Enhancing Electricity Audits in Residential Buildings with Nonintrusive Load Monitoring." In: 2010.

[4] M. Bergés, E. Goldman, H. S. Matthews, and L. Soibelman. "Learning Systems for Electric Consumption of Buildings." In: 2010.

[5] L. Breiman. "Random Forests." In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324.

[6] P. G. Christoph Klemenjak. "Non-Intrusive Load Monitoring: A Review and Outlook." In: 1 (2016).

[7] A. Cole and A. Albicki. "Nonintrusive identification of electrical loads in a three-phase environment based on harmonic content." In: *Proceedings of the 17th IEEE Instrumentation and Measurement Technology Conference [Cat. No. 00CH37066]*. Vol. 1. May 2000, 24–29 vol.1. DOI: 10.1109/IMTC.2000.846806.

[8] L. B. Du, Y. Yang, D. He, R. G. Harley, T. G. Habetler, and B. Lu. "Support vector machine based methods for non-intrusive identification of miscellaneous electric loads." In: *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society* (2012), pp. 4866–4871.

[9] M. B. Figueiredo, A. de Almeida, and B. Ribeiro. "An Experimental Study on Electrical Signature Identification of Non-Intrusive Load Monitoring (NILM) Systems." In: *ICANNGA*. 2011.

[10] J. Gao, S. Giri, E. C. Kara, and M. Berges. "PLAID: a public dataset of high-resoultion electrical appliance measurements for load identification research: demo abstract." In: *BuildSys@SenSys*. 2014.

[11] P. Geurts, D. Ernst, and L. Wehenkel. "Extremely randomized trees." In: *Machine Learning* 63.1 (Apr. 2006), pp. 3–42. DOI: 10.1007/s10994-006-6226-1.

[12] G. W. Hart. "Nonintrusive appliance load monitoring." In: *Proceedings of the IEEE* 80.12 (Dec. 1992), pp. 1870–1891. ISSN: 0018-9219. DOI: 10.1109/5.192069.

[13] K. He, L. Stankovix0107, J. Liao, and V. Stankovix0107. "Non-Intrusive Load Disaggregation Using Graph Signal Processing." In: *IEEE Transactions on Smart Grid* 9 (2018), pp. 1739–1747.

[14] P. Heracleous, P. Angkititrakul, N. Kitaoka, and K. Takeda. "Unsupervised energy disaggregation using conditional random fields." In: *IEEE PES Innovative Smart Grid Technologies, Europe* (2014), pp. 1–5.

[15] A. K. Jain, S. S. Ahmed, P. Sundaramoorthy, R. Thiruvengadam, and V. Vijayaraghavan. "Current peak based device classification in NILM on a low-cost embedded platform using extra-trees." In: *2017 IEEE MIT Undergraduate Research Technology Conference (URTC)* (2017), pp. 1–4.

[16] M. J. Johnson and A. S. Willsky. "Bayesian nonparametric hidden semi-Markov models." In: *Journal of Machine Learning Research* 14 (2013), pp. 673–701.

[17] I. T. Jolliffe. "Principal Component Analysis." In: *International Encyclopedia of Statistical Science*. 2011.

[18] D. Jorde, T. Kriechbaumer, and H. Jacobsen. "Electrical Appliance Classification using Deep Convolutional Neural Networks on High Frequency Current Measurements." In: *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. Oct. 2018, pp. 1–6. DOI: 10.1109/SmartGridComm.2018.8587452.

[19] M. Kahl, A. U. Haq, T. Kriechbaumer, and H.-A. Jacobsen. "A Comprehensive Feature Study for Appliance Recognition on High Frequency Energy Data." In: *e-Energy*. 2017.

[20] M. Kahl, A. U. Haq, T. Kriechbaumer, and H.-A. Jacobsen. "WHITED-A Worldwide Household and Industry Transient Energy Data Set." In: 2016.

[21] T. Kato, H. S. Cho, D. Lee, T. Toyomura, and T. Yamazaki. "Appliance Recognition from Electric Current Signals for Information-Energy Integrated Network in Home Environments." In: *ICOST*. 2009.

[22] J. Kelly and W. J. Knottenbelt. "Neural NILM: Deep Neural Networks Applied to Energy Disaggregation." In: *BuildSys@SenSys*. 2015.

[23] J. Kelly and W. J. Knottenbelt. "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes." In: *Scientific data*. 2015.

[24] H. Kim, M. Marwah, M. F. Arlitt, G. Lyon, and J. Han. "Unsupervised Disaggregation of Low Frequency Power Measurements." In: *SDM*. 2011.

[25] J. Z. Kolter. "REDD : A Public Data Set for Energy Disaggregation Research." In: 2011.

[26] J. Z. Kolter and T. S. Jaakkola. "Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation." In: *AISTATS*. 2012.

[27] J. H.-A. Kriechbaumer Thomas. "BLOND, a building-level office environment dataset of typical electrical appliances." In: *Scientific Data* 5 (2018). DOI: 10.1038/sdata.2018.48.

[28] K. Kumar and M. G. Chandra. "Event and feature based electrical load disaggregation using graph signal processing." In: *2017 IEEE 13th International Colloquium on Signal Processing its Applications (CSPA)* (2017), pp. 168–172.

[29] Y.-X. Lai, C.-F. Lai, Y.-M. Huang, and H.-C. Chao. "Multi-appliance recognition system with hybrid SVM/GMM classifier in ubiquitous smart home." In: *Inf. Sci.* 230 (2013), pp. 39–55.

[30] H. Y. Lam, G. S. K. Fung, and W. K. Lee. "A Novel Method to Construct Taxonomy Electrical Appliances Based on Load Signaturesof." In: *IEEE Transactions on Consumer Electronics* 53 (2007).

[31] S. B. Leeb, S. R. Shaw, and J. L. Kirtley. "Transient Event Detection in Spectral Envelope Estimates for Nonintrusive Load Monitoring." In: 2004.

[32] J. Liang, S. S. Y. Ng, G. E. Kendall, and J. W. M. Cheng. "Load Signature Study—Part I: Basic Concept, Structure, and Methodology." In: *IEEE Transactions on Power Delivery* 25 (2010), pp. 551–560.

[33] G.-y. Lin, S.-c. Lee, J. M. Hsu, and W.-r. Jih. "Applying power meters for appliance recognition on the electric panel." In: *2010 5th IEEE Conference on Industrial Electronics and Applications* (2010), pp. 2254–2259.

[34] M. Marceau and R. Zmeureanu. "Nonintrusive load disaggregation computer program to estimate the energy consumption of major end uses in residential buildings." In: *Energy Conversion and Management* 41.13 (2000), pp. 1389–1403. ISSN: 0196-8904. DOI: https://doi.org/10.1016/S0196-8904(99)00173-9.

[35] J. A. Mueller, A. Sankara, J. W. Kimball, and B. M. McMillin. "Hidden Markov models for nonintrusive appliance load monitoring." In: *2014 North American Power Symposium (NAPS)* (2014), pp. 1–6.

[36] L. K. Norford and S. B. Leeb. "Non-intrusive electrical load monitoring in commercial buildings based on steady-state and transient load-detection algorithms." In: 2000.

[37] A. Ocneanu. "Unsupervised disaggregation of appliances using aggregated consumption data." In: 2011.

[38] S. Pattem. "Unsupervised Disaggregation for Non-intrusive Load Monitoring." In: *2012 11th International Conference on Machine Learning and Applications* 2 (2012), pp. 515–520.

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

[40] A. G. Ruzzelli, C. Nicolas, A. Schoofs, and G. M. P. O'Hare. "Real-Time Recognition and Profiling of Appliances through a Single Electricity Sensor." In: *2010 7th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)* (2010), pp. 1–9.

[41] S. R. Safavian and D. A. Landgrebe. "A survey of decision tree classifier methodology." In: *IEEE Trans. Systems, Man, and Cybernetics* 21 (1991), pp. 660–674.

[42] K. Suzuki, S. Inagaki, T. Suzuki, H. Nakamura, and K. Ito. "Nonintrusive appliance load monitoring based on integer programming." In: *2008 SICE Annual Conference* (2008), pp. 2742–2747.

[43] K. Ting, M. Lucente, G. S. Fung, W. Lee, and S. Hui. "A Taxonomy of Load Signatures for Single-Phase Electric Appliances." In: *IEEE PESC (Power Electronics Specialist Conference)* (2005), pp. 12–18.

[44] "Using a pattern recognition approach to disaggregate the total electricity consumption in a house into the major end-uses." In: *Energy and Buildings* 30.3 (1999), pp. 245–259. ISSN: 0378-7788. DOI: https://doi.org/10.1016/S0378-7788(99)00007-9.

[45] M. Zeifman and K. Roth. "Disaggregation of home energy display data using probabilistic approach." In: *2012 IEEE International Conference on Consumer Electronics (ICCE)* (2012), pp. 630–631.

[46] M. Zhong, N. H. Goddard, and C. A. Sutton. "Signal Aggregate Constraints in Additive Factorial HMMs, with Application to Energy Disaggregation." In: *NIPS*. 2014.

[47] T. Zia, D. Bruckner, and A. A. Zaidi. "A hidden Markov model based procedure for identifying household electric loads." In: *IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society* (2011), pp. 3218–3223.

[48] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar. "Non-Intrusive Load Monitoring Approaches for Disaggregated Energy Sensing: A Survey." In: *Sensors*. 2012.