Dissertation zur Erlangung des Doktorgrades der
Technischen Fakultät der
Albert-Ludwigs-Universität Freiburg

Acquiring, Congregating, and Processing High-Frequency Electricity Data in Distributed Environments

Benjamin Völker

2021

Albert-Ludwigs-Universität Freiburg
Technische Fakultät
Institut für Informatik

Dekan
Prof. Dr. Roland Zengerle

Referenten
Prof. Dr. Bernd Becker
Prof. Dr. Kristof Van Laerhoven
Prof. Dr. Christoph Scholl

Datum der Promotion
17.12.2021

Contents

Zusammenfassung

Während Häuser heutzutage immer besser isoliert werden um den Verlust von Wärmeenergie gering zu halten, ist der Energiebedarf durch die Nutzung von Elektrizität in den letzten Jahrzehnten trotz energieeffizienterer Technik gestiegen. Dies ist mitunter auf das Nutzungsverhalten der Verbraucher zurückzuführen. Um dieses Verhalten nachhaltig zu beeinflussen, können *Eco Feedback* oder *Gamification* Ansätze - optimiert auf den Elektrizitätsverbrauch - eingesetzt werden. Non-Intrusive Load Monitoring (NILM) liefert eine retrospektive und leicht zu installierende Möglichkeit, den Gesamtenergieverbrauch nach den einzelnen Geräten aufzuschlüsseln. Dies hilft einen zu hohen Energieverbrauch einzelner Haushaltsgeräte zu identifizieren und verbessert Eco Feedback und Gamification Methoden.

Um NILM Systeme und Algorithmen weiterzuentwickeln und zu vergleichen, werden spezielle Datensätze benötigt, bei denen neben dem Gesamtverbrauch auch der tatsächliche Energiebedarf aller Verbraucher im Messsetup erfasst wurde. Außerdem können zusätzliche Informationen über die Zustände (vereinfacht zB. *angeschaltet* oder *ausgeschaltet*) der einzelnen Geräte von Vorteil sein. Da NILM Systeme häufig ein aufwendiges, systemspezifisches Training erfordern, können sie bislang lediglich als Nischenprodukt angesehen werden. In der vorliegenden Arbeit werden verschieden Techniken vorgestellt, um detaillierte NILM Datensätze zu generieren. Außerdem werden Methoden untersucht, um die aufwendigen Trainingsphasen, welche nach der Installation solcher Systeme benötigt werden, zu vereinfachen.

Dazu werden zunächst die Anforderungen an ein Messsystem, welches einen solchen Datensatz aufnehmen kann, herausgearbeitet. Des Weiteren wird ein Framework vorgestellt, welches diese Anforderungen nachweislich erfüllt. Das System kann zeitsynchonisierte, hochfrequente Elektrizitätsmesswerte über einen langen Zeitraum aufnehmen, auch wenn die einzelnen Messpunkte über eine größere Flache (zB. ein Haus oder eine Fabrik) verteilt sind und Daten größtenteils kabellos übertragen werden müssen. Das Framework umfasst zwei unterschiedliche Messsysteme, welche für die Messung einerseits einzelner Geräte und andererseits des Gesamtverbrauchs optimiert wurden.

Um den Informationsreichtum der aufgenommenen Daten zu verbessern, wurde ein Algorithmus entwickelt, der Veränderungen des Energiebedarfs automatisch gruppiert und annotiert. Eine Evaluation des Algorithmus auf zwei Datensätzen hat

gezeigt, dass je nach Verbrauchertyp ein Großteil dieser Veränderungen erfolgreich als solche erkannt und gruppiert werden können, was zu einer erheblichen Zeitersparnis von bis zu 74 % bei der Annotation von Elektrizitätsdaten führen kann. Um die Annotation weiter zu vereinfachen und eine geführte, halbautomatische Annotation der Daten zu ermöglichen, wurde ein entsprechendes Tool entwickelt, in dem der Algorithmus integriert ist.

Mit Hilfe des Frameworks und des Annotiationstools wurde schließlich der Fully-labeled hIgh-fRequency Electricity Disaggregation (FIRED) Datensatz aufgezeichnet. Dieser enthält über 100 Tage hochauflösende Strom- und Spannungsdaten des Gesamtelektrizitätsverbrauchs eines Haushaltes. Neben dem Gesamtverbrauch wurden ähnlich hochauflösende Strom- und Spannungsdaten von 21 Haushaltsgeräten gesammelt und zeitlich synchronisiert. Außerdem wurden weitere Umgebungsgrößen wie Raumtemperatur und Luftfeuchte gemessen. Zwei Wochen dieser Daten wurden mit Hilfe des entwickelten Tools vollständig annotiert.

Es wurde weiter analysiert wie effektiv unterschiedliche domänenspezifische Features und Klassifizierer verschiedene Haushaltsgeräte voneinander unterscheiden können. Die Features und Klassifizierer wurden so gewählt, dass sie auf eingebetteten Systemen mit beschränkten Rechenkapazitäten, wie Smart Metern, eingesetzt werden können. Des Weiteren wurden drei Methoden evaluiert, die in Gesamtverbrauchsdaten Geräte-Zustandsübergänge von den konstanten Beiträgen anderer Geräte bereinigt. Da Gesamtverbrauchsdaten mehr Rauschen beinhalten können, wurde untersucht, ob die Effektivität entsprechender Klassifikationsalgorithmen steigt, wenn während des Trainings zusätzlich auf individuelle Messdaten zurückgegriffen wird. Schließlich wurde ein System vorgestellt, welches neben eines typisches NILM Aufbaus noch weitere Messsysteme auf Steckerebene verwendet um den Trainingsprozess zu vereinfachen und die Gesamtperformanz des Systems zu verbessern.

Abstract

The energy consumption of a home depends on the behavior of its inhabitants offering a promising energy saving potential. However, this potential can only be unfolded to full extent if the consumption of each individual appliance is know. Non-Intrusive Load Monitoring (NILM) offers a retrospective way to get individual appliance consumption data. If such data are combined with eco-feedback techniques it can help to better understand a user's electricity usage to ultimately save energy.

Researching NILM algorithms, and in particular, the development of the underlying supervised machine learning techniques, requires adequate datasets with corresponding ground truth data and methodologies to create more labeled data when needed. Adding detailed labels to such datasets is a time-consuming and error-prone process. Deploying a supervised NILM system typically requires a dedicated system training procedure hampering their widespread adoption. The thesis at hand presents several strategies to address these challenges in order to improve the adoption of NILM.

In particular, a set of requirements is presented for acquiring and congregating high-frequency electrical measurements in distributed environments. These are handled by a novel recording framework comprised of a central recording director and two prototype Data Acquisition Systems (DAQs), one for aggregated and one for plug-level data. The developed methodologies allow the DAQs to deliver highly accurate and time-synchronized data while using rather inexpensive components.

To add precise and descriptive labels to such data, a semi-automatic labeling method is developed and evaluated on two publicly available datasets. The method improves the labeling efficiency up to 74 % and has been integrated into a novel labeling tool implemented as a web-application.

The framework and labeling tool have been used to collect and label the Fully-labeled hIgh-fRequency Electricity Disaggregation (FIRED) dataset. It contains 101 days of 8 kHz aggregated current and voltage measurements of the 3-phase electricity supply of a typical residential apartment in Germany. The data also includes synchronized 2 kHz plug-level readings of 21 individual appliances, other environmental sensor measurements, and descriptive event labels of all appliances, resulting in a complete and versatile residential electricity dataset.

Furthermore, several domain specific features and classifiers are evaluated regarding their suitability for (event-based) NILM targeted for resource constrained systems. Data cleaning methods are evaluated which remove the steady-state energy consumption of other appliances from the aggregated data of a given appliance event. As plug-level data delivers less noisy individual appliance data, it is shown that the inclusion of such data during training results in a performance gain for appliance classification algorithms. Finally, a novel supervised NILM system is proposed and evaluated which uses a combination of aggregated and individual appliance data to improve and aid the training process while only requiring minimal user interaction.

1 Introduction

The anthropogenic climate change has caused and will continue to cause a global temperature change in the coming years. Electricity production, heating, and transportation are the main contributors to greenhouse gas emissions (CO_2) with approximately 73.2 % [1] (residential buildings account for 11 %). Globally, approximately 27 % of the produced electricity is used to power lighting, appliances, and heating in the residential sector [2]. Still, only 25.6 % of this electricity is produced by carbon-neutral generation methods (nuclear power excluded) [2].

At the time of writing and also during the last decades, the price for a kilowatt-hour of electricity is comparably low (around 0.30€ in Germany [3]) resulting in a relatively low incentive to save electricity. However, with rising energy costs - indeed the price has increased constantly over the last decades - this attitude can be assumed to change resulting in higher incentives to identify and replace energy-hungry appliances.

As famously stated by Peter Drucker: "If you can't measure it, you can't improve it". Electrical energy is measured with electricity meters. If such electricity measurements are available, they can be combined with eco-feedback techniques. These techniques range from simple graphs showing current and historical consumption data [4], ambient installations in electricity cords [5], or artistic environmental installations (*7000 oaks and counting* [6]). Eco-feedback has proven to achieve high energy savings especially in the residential domain according to several studies. A meta-study by Ehrhardt-Martinez et al. [7] found that real-time aggregated-level electricity consumption feedback achieves energy savings of 8.6 % on average. If this feedback is delivered for individual appliances, the saving potential can even be increased to 13.7 % on average. This was confirmed by Kelly and Knottenbelt [8], and Serrenho et al. [9]. The latter discovered a 5 to 10 % relative increase if eco-feedback is provided with individual appliance consumption data.

If these findings are representatively related to Germany with approximately 41 million households that consume around 3113 kW h on average per household and year, around 17.8 GW h less electrical energy would be required per year [10]. This is the equivalent production capacity of one to two medium-size nuclear power plants.

This calculation assumes *standard* eco-feedback which raises the awareness of the electricity consumption by pinpointing the resident to an unnecessary consump-

tion. If such feedback is combined with a smart home agent which can directly control electrical appliances or recommend user-specific strategies, even higher savings are conceivable.

In order to obtain appliance-specific consumption, it is possible to attach a dedicated electricity meter to each consumer of interest or to replace an appliance by a *smart appliance* which directly measures its consumption. This method is called Intrusive Load Monitoring (ILM) due to its *intrusive* nature of having to re-wire or exchange appliances.

A second method to obtain appliance-level data, which does not require to replace existing appliance or infrastructure, is Non-Intrusive Load Monitoring (NILM). NILM requires just a single electricity meter (such as a smart meter) to be installed at a central position in a home (typically inside the fuse box). The measured composite load is disaggregated into the load of each individual electrical consumer in the home using advanced algorithms, utilizing machine learning and pattern recognition methods. These algorithms are either executed on the meter itself, or the data are sent to an external processing system.

Regardless of how appliance-level data are generated (ILM or NILM), privacy concerns need to be considered if such data are processed by or transmitted to external entities. The main drawbacks of NILM are that it only delivers estimated consumption data of individual appliance and typically requires initial training, since most of the underlying algorithms are of supervised nature (i.e., require a training phase for which disaggregated data are already available).

Nevertheless, depending on the employed method, NILM can provide information such as (1) the consumption amount, (2) the consumption pattern, and (3) the current state of an appliance. All of this information can be used, besides eco-feedback, to:

- identify connected appliances or appliance types in an electrical power grid
- identify the state of an appliance and the transition between states (e.g., when a device is switched-on)
- optimize electricity in smart grids using additional demand response techniques
- identify malfunctioning appliances or appliances which require a service (aka predictive maintenance)
- enable Ambient Assisted Living (AAL) if the technique is combined with a smart home agent
- enhance the electricity bill by breaking it down into individual consumers

Due to its physical foundation based on the *flow* of electricity, the corresponding NILM algorithms can also be applied to water and gas consumption covering all

types of energy consumption in our homes. Nevertheless, this thesis representatively covers the electrical domain.

In many countries, smart meters have been and will be gradually installed in homes in the upcoming years. These meters allow to collect the electrical power consumption at more fine-grained spatial and temporal resolution compared to previous rotary-disc meters. In addition, network interfaces allow to transmit these data directly to the electricity provider which simplifies the billing of electrical energy and enables electricity data analytics at scale [J21b]. Despite that, the shift towards smart meters allows NILM algorithms to utilize not only the smart meters' data but also their processing capabilities.

In fact, some companies and electricity providers already offer services based on NILM (e.g., *Discovergy* [11] and *sense* [12]) due to the benefits obtained by appliance-level data. Eco-feedback and energy saving recommendations have also been tested by major players such as Google [13] or Microsoft [14] but were eventually discontinued due to only moderate interest amongst potential customers. This again underscores the need to increase and maintain a constant incentive to save electricity.

1.1 Problem Formulation

Despite the intended benefits, NILM methods still suffer from obtrusive training and mediocre disaggregation performance [15] if electricity of more than a handful of appliances is disaggregated. One reason for that is a lack of suitable datasets. This work mainly targets challenges during the following steps: data(set) acquisition, dataset labeling, and supervised appliance classification.

Data(set) acquisition: To train, evaluate and compare NILM algorithms, publicly available datasets are used. Even though a lot of datasets have been published, they are often not suitable to compare different disaggregation techniques because of a low sampling frequency (e.g., 1/60 Hz for AMPds [16]), large recording gaps (e.g., 13 of 37 days for REDD [17]), or because of missing or incorrect ground truth data. Recording new datasets is an expensive, obtrusive, and tedious endeavor requiring knowledge in electrical engineering as well as signal and data processing.

Dataset labeling: Moreover, most datasets lack ground truth information (such as appliance events) making them hardly applicable to algorithms without additional ground truth generation. Manually adding such ground truth data (i.e., appliance event labels) post recording is a time-consuming, tedious, and error-prone

process. While many event detection algorithms are suitable to automate this process, they are typically based on simple rules (e.g., predefined thresholds) which might work well for classical electrical appliances. Nowadays, most consumer electronics devices are powered by Switched-Mode Power Supplys (SMPSs), such as TVs, laptops, etc., resulting in heterogeneous power consumption with short periods of higher consumption (also called peaks). If an event detector based on simple rules (as e.g. used in [18]) is applied to such data, these peaks lead to many events which are not of interest to the user.

Appliance classification: The results of appliance classification algorithms highly vary depending on the algorithm type, the environment the algorithm is applied in, the data applied to it, and the appliance usage patterns. Many different approaches exist either based on simple rules, feature-driven machine learning algorithms, or deep neural networks. Compared to unsupervised or semi-supervised methods, supervised methods typically show significantly better results after being trained on the target home ($F_1 > 0.9$ e.g., [19] and [20]). To obtain the required labeled data, however, an intrusive (i.e., tedious and time-consuming) training is indispensable before the system can be used in a non-intrusive fashion. While training, the home owners has to switch appliances *on* and *off* multiple times with different, preferably all possible combinations of concurrently running appliances, taking the word *non-intrusive* ad absurdum. To address this problem, new strategies need to be explored e.g., by taking the human into the loop of the machine learning process.

1.2 Systematic Approach

At first, this work states a set of requirements for electricity metering systems targeted to record advanced electricity datasets for NILM. Such a system is implemented and the integrated hardware and software components are described in detail. This system serves as a framework to obtain aggregated-level and individual appliance-level data in distributed environments. Second, semi-automatically labeling methods are researched that allow to add fully-labeled state changes of all appliances to electricity datasets in a post-processing fashion. Third, the framework and labeling is applied and evaluated by recording a novel long-term electricity dataset which features fully-labeled aggregated and individual appliance data measured at high sampling rates. Lastly, strategies are explored and evaluated to improve the lengthy training procedure and overall performance of supervised event-based NILM methods by using additional measurement devices and including humans in the loop.

Recording framework: A set of requirements for a recording framework is defined and hardware and software components, which meet these requirements, are

presented. The developed framework includes two newly engineered Data Acquisition Systems (DAQs): the *SmartMeter* and the *PowerMeter*. The SmartMeter is tailored to record the aggregated electricity consumption from inside a home's fuse box. The meter can monitor up to three supply legs (i.e., current-carrying lines in the electricity grid) with up to 32 kHz. The overall system architecture of the PowerMeter is comparable to the SmartMeter. However, the PowerMeter is tailored to measure individual appliances directly at plug-level with up to 7.875 kHz in a distributed environment. By adding modular design concepts, both meters remain expandable while keeping the costs of the setups comparably low (around €35 for a PowerMeter and €100 for a SmartMeter). Both meters are integrated into a software architecture that synchronizes all meters, processes raw voltage and current measurements, converts them into physical quantities, and stores them into files. Therewith, the framework can record long-term continuous datasets dedicated to evaluate a large set of electricity-related algorithms such as NILM.

Data labeling: Ways to automatically tag electricity data with fine-grained event labels are explored by using an event detection algorithm, clustering, and human supervision. As classical event detection algorithms do not perform well for appliances powered by SMPS, a novel probabilistic event detector with adaptive thresholding is developed. To detect reoccurring events, an unsupervised clustering method clusters and pre-labels events which are afterwards filtered to remove false events in noisy signal portions. The algorithm is integrated into a novel labeling tool named Annoticity. Annoticity is designed to provide a simple Graphical User Interface (GUI) to enable human supervision for automatically generated labels. It further offers convenient visual access to many publicly available datasets.

The FIRED dataset: A novel residential electricity dataset called FIRED is presented. The specific environment of FIRED as well as the data collection procedure is outlined. The previously introduced recording framework has been used to record long-term electricity data at aggregated level and of 21 individual appliances. The collected data includes 101 days of continuous voltage and current waveforms sampled with 8 kHz and 2 kHz, respectively. Additional sensor measurements such as room temperature and lighting states further augments these recordings. The previously introduced Annoticity labeling tool has been used to add two weeks of fine grained labels to the data.

Appliance classification: Different appliance event detection algorithms are examined and evaluated. Afterwards, a benchmark set of events is extracted from the public datasets FIRED, BLOND [21], WHITED [22], and PLAID [23]. The set is used to evaluate 27 features on their ability to classify appliances. This information is used to infer a suitable feature vector resulting in high classification results while maintaining low dimensionality. The feature vector is used to evaluate

four standard machine learning classifiers. The combined system (encompassing the event detector, the feature extraction, and the classifier) acts as a benchmark NILM system. In addition, the combined training on high-frequency electricity data sampled at aggregated and appliance-level is investigated. The results show slight but noticeable improvements over the previously determined benchmark system.

Minimal-Intrusive Load Monitoring: It was also investigated how the training process for supervised NILM systems can be supported and accelerated by the use of (intrusive) plug meters. The approach is termed Minimal-Intrusive Load Monitoring (MILM) to hint towards the fact that the training process is optimized towards a minimal and practical effort. The plug meters are used in an adaptive training approach, which determines a minimal set of meters required to boost overall system performance to a desired level before prompting a home owner to change the meter configuration.

1.3 Contribution

This work contributes to (1) the acquisition of electricity data for research purpose, (2) semi-automatic appliance event labeling, (3) supervised appliance classification, and (4) the evaluation of NILM systems by introducing the FIRED dataset and the Annoticity labeling tool.

The main contribution regarding the acquisition of electricity data are:

(i) A new data acquisition system to measure the aggregated load of a home is proposed. The system's architecture allows flexible ways to incorporate NILM algorithms either on the device itself or by passing the data via different interfaces to an external data-processing unit.

(ii) A new data acquisition system to measure the electrical energy consumption of individual appliances is proposed. The system samples measurements at rates of up to multiple kilosamples per second, which allows to analyze higher frequency components in the data.

(iii) An overall framework for collecting high-frequency (electricity) data in distributed environments is presented, tailored for time-synchronized data, high robustness, expandability, and simplified usability.

The main contribution regarding semi-automatic appliance event labeling are:

(i) A novel probabilistic event detection algorithm is proposed and evaluated. The detector is combined with adaptive thresholding to suppress false positives for low power appliances.

(ii) Additional clustering techniques are used to find reoccurring events in a continuous data stream allowing to tag each event with a corresponding label.

(iii) These techniques are encompassed into semi-automatic labeling. Techniques are being developed to manually label high-frequency electricty data quickly and easily.

The main contribution regarding (supervised) appliance classification are:

(i) 27 features are presented and evaluated in a standalone and combined feature analysis. The active power, phase-angle, Tristimulus, and WaveForm Approximation have been identified to be of choice for resource-constrained systems.

(ii) Furthermore, four standard machine learning classifiers are examined for their suitability regarding appliance classification. k-Nearest Neighbour is found to be the algorithm of choice if used on resource-constrained systems while Support Vector Machines and Random Forest should be considered otherwise.

(iii) The concept of hybrid training (i.e., using additional high-frequency data obtained by plug-level sensors during training) is examined which can lead to an F_1-score gain of up to 4 % compared to traditional training methods.

(iv) A concept is developed to enhance the tedious training process and overall performance of supervised NILM systems by using additional plug-level meters.

The main contribution regarding the evaluation of NILM systems are:

(i) A novel fully-labeled electricity dataset is introduced which allows to evaluate a wide variety of electricity-related algorithms (including appliance event detection algorithms).

(ii) Dataset labeling is improved by reducing the overall labeling time up to 74 % and providing additional support if textual labels are required by incorporating clustering methods.

(iii) An appliance classifier is proposed that achieves an average F_1-score of 98 % on four publicly available datasets. The classifier can be used as a benchmark when comparing the performance of similar systems.

Most of the stated contributions have been published in the following original publications:

[J21a] B. Völker, M. Pfeifer, P. M. Scholl, and B. Becker. "A Framework to Generate and Label Datasets for Non-Intrusive Load Monitoring". *Energies* 14.1 (2021): 75.

[J21b] B. Völker, A. Reinhardt, A. Faustine, and L. Pereira. "Watt's up at Home? Smart Meter Data Analytics from a Consumer-Centric Perspective". *Energies* 14.3 (2021): 719.

[C19a] B. Völker, P. M. Scholl, and B. Becker. "Semi-Automatic Generation and Labeling of Training Data for Non-intrusive Load Monitoring". 2019, *Proceedings of the 10th ACM International Conference on Future Energy Systems (e-Energy '19)*, ACM, Phoenix, USA.

[C19b] B. Völker, M. Pfeifer, P. M. Scholl, and B. Becker. "A Versatile High-Frequency Electricity Monitoring Framework for Our Future Connected Home". 2019, *EAI International Conference on Sustainable Energy for Smart Cities (SESC '19)*, Springer, Braga, Portugal.

[N20] B. Völker, M. Pfeifer, P. M. Scholl, and B. Becker. "FIRED: A Fully-labeled hIgh-fRequency Electricity Disaggregation Dataset". 2020, *International Conference on Systems for Energy-Efficient Built Environments (BuildSys '20)*, ACM, Virtual Event, Yokohama, Japan.

[W20] B. Völker, M. Pfeifer, P. M. Scholl, and B. Becker. "Annoticity: A Smart Annotation Tool and Data Browser for Electricity Datasets". 2020, *5th Workshop on Non-Intrusive Load Monitoring (NILM 20)*, ACM, Virtual Event, Yokohama, Japan.

[PA21] B. Völker, M. Pfeifer, F. Wolling, P. M. Scholl and B. Becker. "Introducing MILM - A Hybrid Minimal-Intrusive Load Monitoring Approach". 2021, *Proceedings of the 12th ACM International Conference on Future Energy Systems (e-Energy '21)*, ACM, Virtual Event, Torino, Italy.

[PA18] B. Völker, P. M. Scholl, B. Becker. "Towards the Fusion of Intrusive and Non-intrusive Load Monitoring - A Hybrid Approach". 2018, *Proceedings of the 9th International Conference on Future Energy Systems (e-Energy '18)*, ACM, Karlsruhe, Germany.

[BC19] P. M. Scholl, B. Völker, B. Becker, and K. V. Laerhoven. "A multi-media exchange format for time-series dataset curation". 2019, *Human Activity Sensing: Corpus and Applications*, Springer International Publishing.

1.4 Remainder

The remainder of this work is structured as follows. Theoretical background to smart metering, Non-Intrusive Load Monitoring (NILM), and machine learning is provided in Chapter 2. A more in-depth list of existing techniques for electricity Data Acquisition Systems (DAQs) and NILM methods can be found in Chapter 3. In Chapter 4, a framework is presented comprised of the hardware and software components required to record advanced electricity datasets. In Chapter 5 a novel semi-automatic labeling method for electricity data is introduced. The framework and labeling method were utilized to collect a dataset which is presented in Chapter 6. An extensive evaluation of features and classifiers used for appliance classification is included in Chapter 7 resulting in a benchmark system. Chapter 8 evaluates different cleaning methods for aggregated data, evaluates hybrid training, and introduces a novel NILM system which simplifies supervised training and improves the overall performance using plug-level meters. Finally, in Chapter 9 concluding remarks and directions for future research are highlighted.

2 Background

This chapter introduces the electrical power grid, general electricity metering principles, the NILM pipeline, and machine learning. It aims for a better understanding and serves as a reference for the upcoming chapters.

2.1 (Smart) Electricity Metering

After the contemporaneous invention of the dynamo by Wheatstone, Siemens, and Varley in 1866-67 [24], electricity could be generated and sold in large quantities. With lighting as one of the first and major application for electricity, it became clear that electricity has to be measured for billing purpose (much like the amount of gas consumed by the previously used gas lamps).

While initial electricity meters were based on simple time measurements (Gardiner, 1872), the electrochemical effect (Edison, 1881), the use of pendulums (Ayrton and Perry, 1881), or electrical motors (Thomson, 1889), Bláthy invented the rotary-disc meter (also known as the *Ferraris meter*) in 1889 [25]. The meter utilizes electromechanical induction to rotate an aluminum disc with a velocity proportional to the product of voltage (V) and current (I). Its working principle is nearly identical to the meters used throughout the 20th century with billions of such devices installed worldwide [26].

However, transferring the actual energy consumption data to e.g., utility companies used to be a labor-intensive manual process. Therefore, digital metering devices began to successively replace electromechanical meters in the late 2000s. These so called *smart meters* allow for the collection of electrical power consumption at much more fine-grained spatial and temporal resolutions and their included or additionally added digital communication interfaces allow to report the collected data directly to the utility company.

As shown in Figure 2.1, an electricity meter is located at the entry-point of a building's or apartment's electrical grid connection. The aggregated consumption of all electrical consumers in the home can be captured at this location. To be able to efficiently transmit power through long power grid lines and to easily

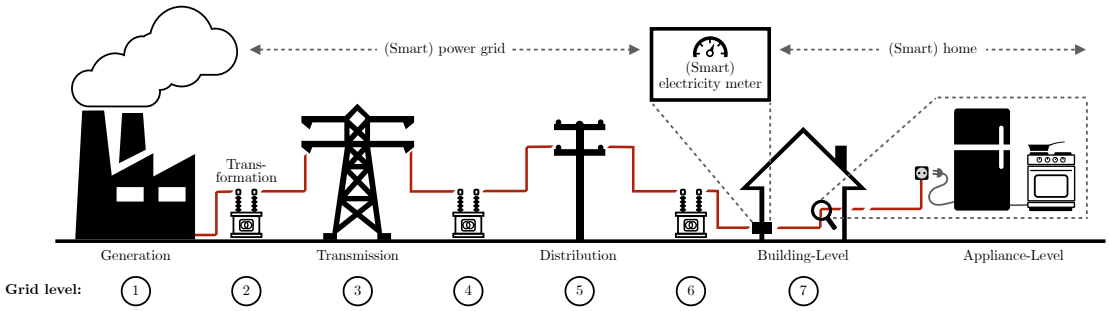

Figure 2.1: Location of (smart) electrical meters within the electrical power grid.

transform it to different voltage levels, electrical energy is distributed as sinusoidal Alternating Current (AC) at high-voltage levels (up to 380 kV) in electrical power grids throughout the world. The voltage level (V_{RMS}) at building-level and the net frequency (f_l) at which the voltage alternates depends on the country. While e.g., the grid in the United States has 110 V_{RMS} at 60 Hz, a standard of 230 V_{RMS} at 50 Hz has been established in the European power grid.

The electrical energy (W) consumed in the time interval $[t_1, t_2]$ is calculated by integrating the Power ($P(t)$) in this interval as

$$W^{[t_1,t_2]} = \int_{t_1}^{t_2} P(t)\,\mathrm{d}t = \int_{t_1}^{t_2} u(t) \cdot i(t)\,\mathrm{d}t. \tag{2.1}$$

Electrical energy is typically billed in *kilowatt hours* (kW/h). While rotary-disc meters are typically read out manually once a year, smart meters store and send the electricity consumption data in the order of seconds to minutes. While their primary use case is billing consumers for the amount of electrical energy consumed, they can also serve as data sources for smart home installations or Building Management Systems (BMS). As smart meters can either directly connect to the Internet or indirectly using smart meter gateways [27], access to metered data is ubiquitously possible. The previously unimaginable temporal resolution of smart meters and their large penetration level, due to the sheer amount of households they are installed in, created the foundation for electricity data analytics and will soon provide novel energy-based services such as NILM [28].

2.2 Non-Intrusive Load Monitoring

Non-Intrusive Load Monitoring (NILM) describes the process of disaggregating a composite electrical load into the load of each individual consumer. Compared to

Intrusive Load Monitoring (ILM), which requires to install a dedicated electricity meter to each consumer, NILM requires just a single electricity meter to be installed in the home's electrical grid (such as a smart meter). The disaggregation process can be based on heuristics, which encode a domain expert's knowledge or machine learning techniques. The algorithms are executed either on the meter itself, or the data are sent to external entities which offer more storage and processing capabilities.

Requiring to install and maintain only a single electricity meter presents the main advantage of NILM compared to ILM. Since a NILM algorithm can ideally be directly applied to smart meter data, the approach is considered to be inexpensive as it allows to retrospectively add individual appliance consumption monitoring using existing infrastructure.

As appliances are connected in parallel to a home's power grid, the current drawn by each appliance (j) sums up at the point where all parallel lines join, according to Kirchhoff's first law [29]. In contrast, the voltage remains constant at each network joint. Therefore, the aggregated momentary power ($P_{sum}(t)$) and thus also the electrical energy consumption (W_{sum}) can be formally written as the sum of the individual appliance power consumption. The equations are given as

$$P_{sum}(t) = \sum_{j=0}^{N-1} i_j(t) \cdot u(t) = \sum_{j=0}^{N-1} P_j(t), \tag{2.2}$$

$$W_{sum}^{[t_1,t_2]} = \int_{t_1}^{t_2} P_{sum}(t)\,\mathrm{d}t, \tag{2.3}$$

with $P_j(t)$ being the power consumption of appliance j and N being the total number of appliances connected to the home's electricity grid.

Therefore, according to Huber et al. [30] the NILM problem can be formally written as

$$P_{sum}(t) = \sum_{i=0}^{M-1} P_i(t) + \sum_{k=0}^{K-1} P_k(t) + \epsilon(t), \tag{2.4}$$

with $P_i(t)$ being the active power consumption of M appliances known to the NILM system (either explicitly modeled or learned during a supervised training phase) and $P_k(t)$ being the active power consumption of appliance k from a set of K unknown appliances. $\epsilon(t)$ models additional noise of the measurement system and is assumed to follow a normal distribution and to be small compared to the terms $\sum P_i$ and $\sum P_k$. The term $\sum P_k$ typically does not follow a Gaussian distribution and accounts for a major portion of $P_{sum}(t)$. [30]

The goal of a NILM system is to estimate $P_i(t)$ for each appliance i over time while only directly measuring $P_{sum}(t)$. The electrical energy consumed by known

appliance i can be calculated for a given time period according to Equation 2.1. Therefore, some NILM systems model regression problems to directly estimate $P_i(t)$ ($P_i(t) = f_i(P_{sum}(t), \beta_i) + \epsilon_i$ with $P_i(t) \in \mathbb{R}$). Another approach is to relax the problem into binary classifications ($s_i(t) = f_i(P_{sum}(t))$ with $s_i(t) \in \{0, 1\}$) and to determine if appliance i is currently switched on or off. After classification, the runtime of appliance i can be extracted (t where $s_i(t) = 1$). While runtime information is already vital for several use-cases (e.g., AAL and human activity recognition (HAR) [31–34]), $P_i(t)$ can still be estimated by multiplying the runtime with the average power consumption ($\overline{P}_i^{on}$) of appliance i.

Independent whether a NILM system uses regression or classification, its general pipeline includes the two steps (1) Data acquisition and (2) Disaggregation as shown in Figure 2.2. Data acquisition is comprised of measuring the required attributes (such as active and reactive power) and performing general pre-processing steps while the disaggregation step is a specially designed and often individually trained algorithm. According to e.g., [35–42] most of the NILM systems that can be found in literature can further be categorized into *event-based* and *event-less* approaches.

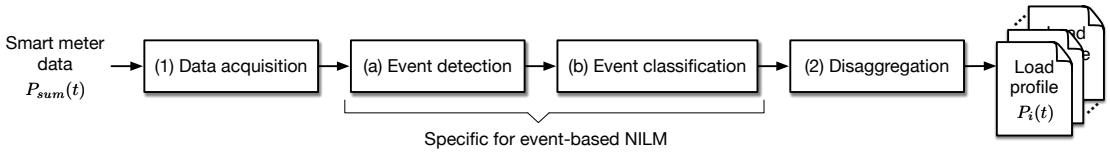

Figure 2.2: General pipeline of event-less and event-based NILM systems. While event-less methods include a data acquisition and a disaggregation step, event-based methods additionally include event detection and event classification.

Event-less non-intrusive load monitoring:
Event-less approaches optimize an overall system state using individually trained appliance models. As the optimization step is recalculated for each new data input, event-less approaches typically suffer from high computational complexity and are often only applied to low sampling rates. Popular event-less approaches are based on Hidden Markov Model (HMM) [43–46] or deep neural networks [47–51].

Event-based non-intrusive load monitoring:
According to Anderson et al. [52] the *event-based* NILM process introduces two additional sub-steps to the pipeline as depicted in Figure 2.2: (a) Event detection and (b) Event classification. After detecting an appliance event, i.e., a state change in the measurements, the event is classified following the pattern matching paradigm. Features are extracted from the measurements and are fed into a classifier, which infers the event class (i.e., the type of the event such as a specific appliance turning

on). As the inference step is only applied to events, and events are assumed to be rare, event-based methods are generally more efficient if only lightweight event detection algorithms are applied. In Section 7.1.2, various features, which have been handcrafted by domain experts, are presented in more detail. These characterize either the transient profile of the event (*transient* features), or the signal after the event (*steady-state* features). In addition, several classifiers are presented in Section 7.1.3. More details on machine learning, including the classification problem, can be found later in this chapter (see Section 2.4).

High- vs. Low-frequency NILM:
NILM systems can be further divided based on the temporal resolution of their input data. Approaches applied to data with a sampling frequency smaller 50 Hz are considered as *low-frequency* approaches, while systems applied to data which has been sampled faster than 1 kHz are considered as *high-frequency* approaches. Armel et al. [53] and others showed that high-frequency NILM allows to identify roughly 30 % more appliances with around 25 % higher accuracy albeit requiring more processing power. If the data cannot be processed locally (e.g., directly on the smart meter) enough bandwidth is required to transmit the data to external data centers.

2.3 Data Acquisition

Data acquisition describes the process of measuring continuous-time, analog signals as discrete-time, digital signals which are later to be analyzed. This includes the conversion into a measurable unit (typically voltage levels), possible amplification or attenuation, filtering, A/D conversion, and further signal pre-processing according to Figure 2.3.

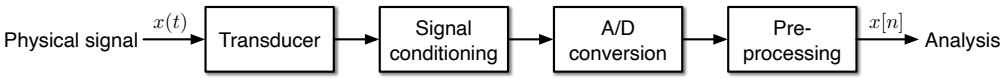

Figure 2.3: General pipeline of a data acquisition. A continuous-time signal $x(t)$ is converted to a discrete-time signal $x[n]$ using a transducer, signal conditioning, an ADC, and pre-processing.

As formally stated in Equation 2.4, the input data of a general NILM system is the home's aggregated power consumption over time. To measure active power, the voltage level and current flow inside the home's electricity grid needs to be measured. The most common methods to measure grid line voltage levels are voltage dividers or voltage transformers. Current flow is typically converted into a

proportional voltage level using shunt resistors, current transformers, hall sensors, or Rogowski coils.

Analog-to-Digital Converters (ADCs) convert these analog voltage levels into digital representations. With sequential A/D conversions, a continuous-time analog voltage signal is, therewith, transformed into a discrete-time digital representation of that signal. Since A/D conversion can lead to aliasing artifacts, analog anti-aliasing filters are added to suppress frequencies above the Nyquist-Frequency (Equation 2.5). If the sampling frequency f_s is chosen twice as high as the maximum analyzed signal frequency, the Nyquist-Shannon sampling theorem holds as shown in Equation 2.6.

$$f_{Nyquist} = \frac{f_s}{2} \tag{2.5}$$

$$f_s > 2 \cdot f_{max\ signal} \tag{2.6}$$

Important quantities of ADCs include their resolution (i.e., number of bits), Signal-to-Noise Ratio (SNR), and maximum sampling rate. While the sampling rate determines the maximum signal frequency that can be reconstructed from the sampled signal according to Nyquist-Shannon, the resolution and SNR determine the minimal and maximal detectable change in voltage.

2.3.1 Data Pre-Processing

The analog frontend can introduce an offset and a phase shift to the measured signal which has to be removed as these may introduce errors in later stages of the processing pipeline. Furthermore, the analog-to-digital conversion step, data caching, or the transmission over certain communication channels raise the possibility of errors and signal falsifications that need to be eliminated. Measurements that do not represent valid number representations and infeasible values (e.g., current flows exceeding the nominal circuit breaker limits by a large factor) are thus removed. Unless a long sequence of wrong data are being reported, the imputation of values and the interpolation of gaps in the sampled data (e.g., by using the *impyute* library [54]) are effective means to prepare the data for further analysis.

Figure 2.4 illustrates 100 ms of voltage ($v[n]$) and current ($i[n]$) measurements of three different electrical loads connected to mains voltage. The signals were sampled at a rate of $f_s = 2000\,\text{Hz}$. A single mains cycle is highlighted in green and refers to a full cycle of the voltage signal which sinusoidally oscillates between $325\,\text{V}$ and $-325\,\text{V}$ ($\equiv 230\,\text{V}_{\text{RMS}}$) with the line frequency $f_l = 50\,\text{Hz}$ in European electricity grids.

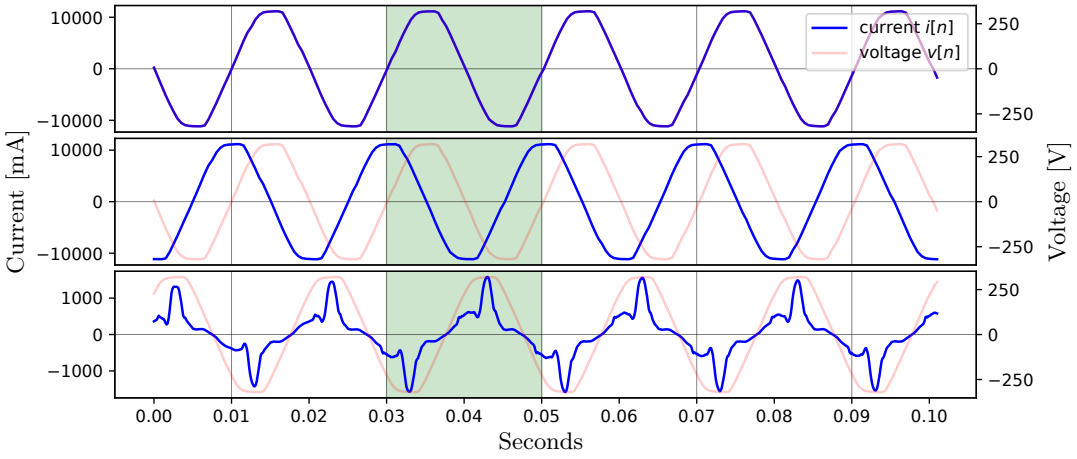

Figure 2.4: Five main cycles of current and voltage measurements with a sampling rate of 2000 Hz. On top, a pure resistive load is shown, in the middle, a pure reactive load is shown, while a typical load containing resistive and reactive components is shown at the bottom. One main cycle is exemplarily highlighted in green.

If a particular measurand is required for the analysis but not directly measured during data acquisition, it needs to be calculated as part of the pre-processing step. In the domain of NILM and energy data analytics, several other measurands besides the sampled voltage and current are of interest. The corresponding formulas to calculate some of these measurands are explained in the following. It is noted that additional measurands and their corresponding formulas are introduced in Section 7.1.2.

To simplify the calculations with AC signals, their corresponding Root Mean Square (RMS) values are often provided. The RMS value of an AC signal is equal to the Direct Current (DC) required to dissipate the same electrical power in a resistive load. I_{RMS} and V_{RMS} can be calculated from the raw voltage and current samples on the basis of a single main cycle (m) as

$$I_{RMS}[m] = \sqrt{\frac{1}{N} \cdot \sum_{n=0}^{N-1} i[n]^2}, \tag{2.7}$$

$$V_{RMS}[m] = \sqrt{\frac{1}{N} \cdot \sum_{n=0}^{N-1} v[n]^2}. \tag{2.8}$$

N is the window length calculated as $N = f_s / f_l$ (with $f_s > f_l$). f_s is the sampling frequency and f_l is the grid line frequency (50 Hz for European countries).

Active ($P[m]$), apparent ($S[m]$), and reactive ($Q[m]$) power can be calculated as

$$P[m] = \frac{1}{N} \cdot \sum_{n=0}^{N-1} v[n] \cdot i[n], \tag{2.9}$$

$$S[m] = I_{RMS}[m] \cdot V_{RMS}[m], \tag{2.10}$$

$$Q[m] = \sqrt{S[m]^2 - P[m]^2}. \tag{2.11}$$

A purely resistive load causes voltage and current to reverse their polarity at the same time as shown in the top plot of Figure 2.4. At every given time instance, the power (cf. Equation 2.1) is positive (or at least zero). For a purely reactive load, voltage and current are phase shifted by 90 degrees and the product of voltage and current is positive only for the first and third quarter of the a main cycle and negative for the second and fourth quarter as shown in the second plot of Figure 2.4. A negative value means that energy stored in capacitors and/or inductors is flowing back from the load. The third plot in Figure 2.4 highlights a typical load with resistive and reactive components. Using Equation 2.9 and Equation 2.11, the active and reactive power can be calculated for a given load. These characterize the loads active (resistive) and reactive (capacitive and inductive) components. The apparent power is the product of the RMS values of the current and voltage signal.

The phase shift between voltage and current ($cos\,\Phi$) can be calculated as

$$cos\,\Phi[m] = \frac{P[m]}{S[m]}. \tag{2.12}$$

The consumed electrical energy can be calculated for each main cycle m or a time period of M main cycles as

$$W[m] = P[m] \cdot \frac{1}{f_l \cdot 3600}, \tag{2.13}$$

$$W_{total} = \sum_{m=0}^{M} W[m]. \tag{2.14}$$

The electrical resistance R and its reciprocal the admittance Y are defines as

$$R[m] = \frac{V_{RMS}[m]}{I_{RMS}[m]}, \tag{2.15}$$

$$Y[m] = \frac{1}{R[m]}. \tag{2.16}$$

If the temporal resolution ($T = 1/f_s$) at which these measures are sampled during data acquisition does not suite the analysis, resampling can be applied. Reducing the rate at which values are being made available, i.e., *downsampling* data, is usually trivial and computationally lightweight as long as the original data has undergone low-pass filtering to avoid aliasing artifacts. Commonly used methods to downsample data include subsampling, averaging, and interpolation [55, 56]. Conversely, increasing the temporal resolution of data is not as trivial, but may be required if the data are reported at very low sampling rates. Interpolation techniques like *Super-Resolution* [57] have shown to achieve good performance on electricity data.

Since the sampling rate mainly determines the amount of data and, therefore, the required bandwidth of the communication channel (in case the data needs to be transmitted) and also the system's processing power, finding the optimal sampling rates for various electricity load analysis algorithms has been investigated in numerous works, including [58–61].

2.4 Machine Learning

Machine learning algorithms use observations from the outside world to generate models that represent a generalized form of the underlying learning task. They are able to find patterns in the data which are then utilized to infer information for yet unseen data. Generally, it is distinguished between *classification* and *regression problems*. A classification problem attempts to find a mapping from the input data to a discrete set of values, typically a string representation aka label. An example is predicting whether a picture shows a cat or a dog. A regression problem attempts to find a mapping to a continuous value, e.g., the price of a house. The corresponding machine learning models are called *classifiers* and *regressors*. Popular application areas of and competitions in machine learning include the detection of objects or creatures in images [62] and the recognition of hand-written numbers [63]. Classical machine learning algorithms require dedicated data pre-processing and feature extraction steps that are typically designed by human experts with specific domain knowledge. Deep learning refers to a kind of a machine learning algorithms that do not require the manual design of a feature extraction step as important features are learned automatically from the raw input data. Machine learning methods can be further categorized into *supervised*, *unsupervised*, and *semi-supervised* methods based on the data they require during model training. Supervised techniques require data with the corresponding ground truth called *labels* (i.e., the data that should be inferred) to adjust the model parameters during training. This requires that labeled data e.g., marked

appliance on-phases are available. Generating these labels may require additional sensors or extensive manual labeling by a human expert. Conversely, unsupervised techniques do not require labels and capture all model parameters based on the target data, which, however, usually results in a lower performance compared to supervised methods. Semi-supervised methods combine a small portion of labeled training data with a typically larger portion of unlabeled data. These methods are mainly beneficial if generating labeled data is a tedious process, while generating unlabeled data is easy. Often, methods are also referred to as semi-supervised if they are trained to generalize their models on data from a certain distribution while the target data are of a different distribution. Related to NILM, this refers to a system being trained on one or multiple homes while being applied on data of a different home which has not been included during system training.

2.5 Model Evaluation

The evaluation of a supervised estimator such as a regressor or classifier is typically performed on data that includes the corresponding ground truth. To compare algorithms against each other, publicly available datasets are used which consist of a set of input data (X) and the corresponding ground truth (Y). An entry (x, y) in a dataset is called a *sample*. If an algorithm should be evaluated on the same dataset it has been trained on, the performance has to be determined on a portion of the dataset that has not been used for training. This is achieved by initially performing a *train-test* split on the data as depicted in Figure 2.5.

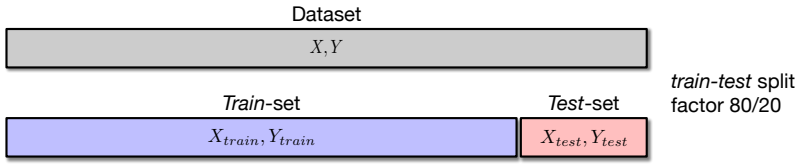

Figure 2.5: Exemplary *train-test* split of a dataset.

The training set (X_{train}, Y_{train}) is used to select an appropriate machine learning model and to optimize its parameters, while the test set (X_{test}, Y_{test}) should only be used once to determine the performance of the final model fitted on the complete training set. A typical train-test split ratio is 80 to 20 % (often written as 80/20).

Using the test set multiple times can lead to *overfitting*. Overfitting denotes that a model has been fitted too close to the input data. While this can lead to perfect results on the test set, it will typical fail on yet unseen data as it prevents the generalization performance of the model.

To decide for an appropriate model (aka *model selection*), a technique called Cross Validation (CV) can be applied which further splits the training set into complementary splits used for training and for model validation (X_{val}, Y_{val}). The final model is then decided upon the best performing model on the validation set as depicted in Figure 2.6.

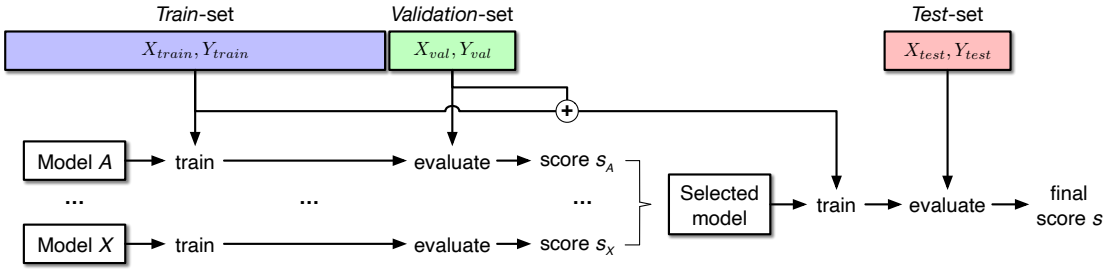

Figure 2.6: Model selection using CV with a *train-validation-test* split.

Fixing the validation set prevents that the complete training set can be exploited during training and may also lead to overfitting. This means that the selection of the model is biased by the performance on the specific validation set. Hence, a different model may have been selected for a different validation split. To compensate this, CV can be applied in several rounds using a different validation split each round. The validation results of all rounds are then combined to provide a more reliable estimate of the model's performance. Several CV methods exist which mainly differ depending on how the splits are performed. Arguably the most frequently used CV method is k-fold CV, which splits the data into k equally sized disjoint splits. $k-1$ splits are used to train the model and the remaining split is used for validation. This is repeated k times, each time choosing a different validation set from the k splits. The results of each round are finally averaged to estimate the overall model performance. 5-fold CV is exemplarily shown in Figure 2.7. The main benefit of this method is that all samples in the data are used for training and validation.

The CV splits can also be performed specifically to test for a certain generalization capability of the models. For instance, if a NILM dataset includes electricity data of different homes, the splits can be performed so that each includes data of a single home only. By averaging the performances over the different CV rounds, the model's ability to generalize across homes is tested.

In many classification problems, some classes naturally occur more often. An example of such an *imbalanced* classification problem is the detection of cancer. Naturally, there are significantly more people without cancer (majority class) than

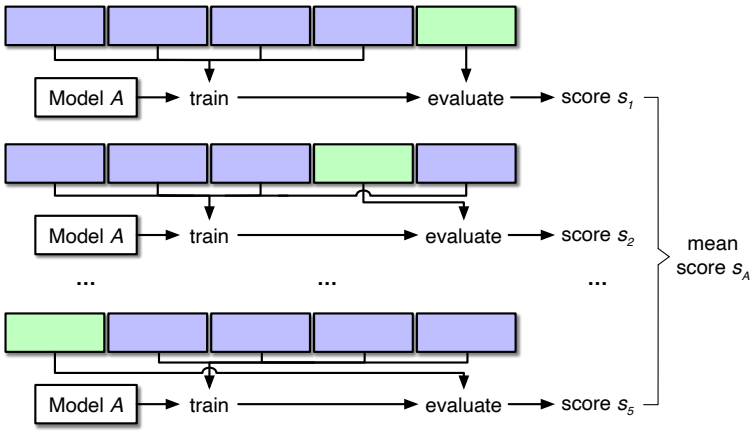

Figure 2.7: k-fold cross validation exemplarily illustrated for $k = 5$.

with cancer (minority class). Datasets for imbalanced classification typically reflect the specific imbalance (e.g., by including 95 % cancer-free, and only 5 % cancer samples). Typically, a correct classification of the majority class is more or at least equally important than of the minority class (for a binary classification). To compensate for this, various techniques can be applied to balance the class distribution of the original dataset. Examples of these techniques include undersampling of the majority class, oversampling (i.e., repetition) of the minority class, or more advanced methods which synthetically generate new samples for the minority class (e.g., ADASYN [64] or SMOTE [65]). However, this is typically not required if the data only contains a *slight* imbalance which represents a ratio of up to 1:4 [66]. Nevertheless, the corresponding CV splits should be performed *stratified* to maintain the same class distribution for each split. Randomly splitting the data can shift this ratio (or even cause splits without a certain class), especially if the number of samples per class is small. This can result in a large bias towards the majority class.

Depending on the used machine learning model, some parameters are not trained during the learning process. These are called *hyperparameter* and can be fixed by the user or tuned during Hyperparameter Optimization (HPO). Since hyperparameters are often of real-valued or unbound space and some algorithms require to tune multiple of these (e.g., around 20 for XGBoost [67]), different techniques can be applied to reduce the search space including random search or Bayesian optimization [68]. It is also possible to define a set of values for each hyperparameter and test all combinations of these values. This technique is called exhaustive grid search.

Depending on the estimator type (regressor or classifier) different performance metrics are used. Regressors can e.g., be evaluated using the Mean-Absolute Error

(MAE) or Root-Mean-Square Error (RMSE) metric. These are defined as

$$MAE = \frac{\sum_{t=0}^{T} |\hat{y}_t - y_t|}{T}, \tag{2.17}$$

$$RMSE = \sqrt{\frac{\sum_{t=0}^{T} (\hat{y}_t - y_t)^2}{T}}. \tag{2.18}$$

$\hat{y}_t$ is the predicted value at time t, y_t is the actual value, and T is the time period over which the metrics are calculated.

In contrast, classifiers are typically evaluated based on several metrics that can be calculated from a confusion matrix. The confusion matrix includes the inferred numbers of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) for each class. Typical metrics that can be derived from these numbers are Accuracy (Acc), Precision (Pre), Recall (Rec), and F_1-score. These are calculated as

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2.19}$$

$$Pre = \frac{TP}{TP + FP}, \tag{2.20}$$

$$Rec = \frac{TP}{TP + FN}, \tag{2.21}$$

$$F_1 = \frac{TP}{TP + 0.5 \cdot (FP + FN)}. \tag{2.22}$$

Since the F_1-score balances precision and recall in a single score, it is often used (1) to state the overall performance of the classifier or (2) to tune the model during model selection (3) or during HPO. All classification metrics are calculated on a per-class basis. Whenever a single score is required, the results of all classes have to be combined. This can be achieved by e.g., calculating the corresponding *macro*- or *micro*-average of the metric. For the micro-average the TP, TN, FP, and FN results of the different classes are summed up before the corresponding metric is calculated. For the macro-average, the metric is calculated for each class and afterwards the overall metric is calculated as the unweighted average. The micro-average should be preferred if the overall classification performance should be reported, whereas the macro-average should be preferred if each class is considered equally important. It is further noted that the macro-average does not account for class imbalances. The performance of the minority and majority class contribute equally important to the metric using macro-averaging.

Depending on the actual use case, further optimization to these metrics or completely different metrics are preferable. According to Figure 2.2, a traditional

NILM-system represents a time-series regression problem (disaggregation step). Metrics besides RMSE that have been used in related works include *error in total energy assigned*, *fraction of total energy assigned correctly*, *normalized error in assigned power*, *mean absolute error*, or *relative error in total energy* [69, 70]. In case of an event-based NILM-system two classification sub-problems are added: event detection and event classification. While event classification seeks to find the corresponding label for a given event and a known set of labels, event detection is a binary classification problem, i.e., it is determined if an event has happened in a certain time window or not. Both classification problems are typically evaluated based on *accuracy*, *precision*, *recall*, and F_1-*score* [69].

3 Related Work

This chapter presents state-of-the-art work regarding Data Acquisition Systems (DAQs) which can be used to collect and process long-term electricity datasets. By introducing several existing datasets, their benefits and drawbacks, the requirements and challenges for a DAQ, best suited for high-frequency long-term recordings, are highlighted. Afterwards, related work in the areas of appliance event detection and dataset labeling is presented. Several techniques to manually or automatically label electricity data are shown. Finally, works regarding supervised appliance event classification are listed.

3.1 Data Acquisition Systems

Kolter and Johnson [17] introduced one of the first recording systems suited to record high-frequency data of a home's aggregated electricity consumption. Their systems can further record low-frequency measurements of individual appliances and was introduced in 2011. To record aggregated data, the system uses an oscilloscope probe (Pico TA041 [71]) to measure voltage of a single phase only and Current Transformers (CTs) with a burden resistor to convert the mains current flow into a measurable voltage level. Both current and voltage signals are fed into *NI-9239* ADCs from National Instruments [72]. The readings are collected by a recording laptop at 15 kHz with a resolution of 24-bit. Off-the-shelf plug-level meters developed by Enmetric [73] are used to record individual appliance data. Several of these connect wirelessly to a bridge which is connected to the home's Internet network. Active power measurements of each outlet are sent to a central server at a rate of 1 Hz. Sub-circuit-level data of certain circuit breakers (e.g., of a hardwired lighting) are acquired using CTs connected to an off-the-shelf electricity meter (eMonitor by Powerhouse Dynamics [74]) with a rate of around one measurement every three seconds. Voltage data are not measured and assumed to be at fixed grid level. Kolter and Johnson's system has been used to record the REDD [17] dataset.

As voltage and current waveforms share lots of similarities with audio data, Kelly and Knottenbelt [75] proposed a system which uses off-the-shelf USB sound cards

with stereo line input in 2015. AC-AC transformer are used to scale down the voltage and split-core CT are used to measure current. The sound cards theoretically allow to sample data up to 44.1 kHz at 32-bit. However, data are downsampled to 16 kHz and padded to 24-bit to significantly reduce the storage requirements. Appliance-level data are sampled using off-the-shelf 433 MHz electricity meter plugs (Eco Manager Transmitter Plugs developed by Current Cost [76]) paired with a self-developed base station. Appliances directly connected to the mains are metered using current clamp meters (Current Cost transmitter [77]) which are read out via the same base station. Due to RF collisions when wirelessly polling for or receiving data, only a comparable low sampling rate of one measurement each six seconds is achieved. Kelly and Knottenbelt's system has been used to record the UK-DALE [75] dataset.

Beckel et al. [47] proposed to use an off-the-shelf smart electricity meter and off-the-shelf smart plugs in 2014. Aggregated data are sampled at 1 Hz from the smart meter (E750 from Landis+Gyr [78]) via its SyM^2 interface and includes several electricity-related metrics such as active power, RMS voltage and current, and the phase shifts of all three supply legs. Active power measurements of individual appliances are sampled using smart plugs (Plugwise [79]) at around 1 Hz. The actual sampling rate varies due to a sequential readout, but the data are resampled to 1 Hz. The system by Beckel et al. has been used to record the ECO [47] dataset.

Instead of using off-the-shelf solutions with their potential drawbacks such as low sampling rates or proprietary protocols, in 2018, Kriechbaumer and Jacobsen [21] proposed a recording setup comprised of a custom-built aggregated meter called CLEAR [80] and multiple custom-built power strip meters called MEDAL [81]. CLEAR uses three hall effect CTs (HAL 50-S from LEM [82]) to sense current on all three supply legs and three AC-AC transformers (VB 3,2/1/6 by BLOCK [83]) to scale down the mains voltage levels. A 16-bit bipolar ADC (Analog Devices AD7656A [84]) is used to sense the six channels (three voltage and three current channels) with up to 250 kHz and a SNR of 87.33 dB. To handle the massive amount of data, a Field Programmable Gate Array (FPGA) (Lattice XO2 7000-HC [85]) is used in addition to a single-board PC (LattePanda [86]). The latter equips the data with a timestamp, compresses, stores, and sends them to a sink via a network connection. MEDAL units share the same overall structure as CLEAR. Each MEDAL unit uses a single AC-AC transformer to scale down the mains voltage level and six individual hall effect sensors to measure current flow through the six outlets of the power strip. The input signals are converted using seven independent unipolar 12-bit ADCs (MCP3201 by Microchip Technology [87]). Data are gathered from the ADCs over a Serial Peripheral Interface (SPI) interface using a microcontroller (ATmega324PA by Microchip Technology [88]). The data are forwarded over a USB serial interface to a single-board PC (Raspberry Pi 3 [89]).

On the PC data are again equipped with timestamps, compressed, stored, and sent to a sink over network. Their setup can sample aggregated data up to 250 kHz, individual appliance data up to 50 kHz, and is robust against network dropouts, which has been proven by recording the BLOND [21] dataset.

3.2 Existing Datasets

Several publicly available datasets, some of which were recorded using the DAQs presented in Section 3.1, are detailed and compared in Table 3.1.

Table 3.1: Comparison of different electricity datasets that already have been used to evaluate NILM algorithms. The systems have been recorded in a residential (res.), office, or laboratory (lab.) environment.

Dataset	Domain	Homes	Aggregated resolution	Appliance resolution	Appli-ances	Duration (days)	Total size
Dataport[a] [90]	res.	669	1/60 Hz	1/60 Hz	0-118	30	1.1 GB
AMPds2 [16]	res.	1	1/60 Hz	1/60 Hz	21	730	2.3 GB
iAWE [91]	res.	1	1 Hz	1–1/6 Hz	19	73	29.2 GB
RAE [92]	res.	2	1 Hz	1 Hz	24	59+72	3.3 GB
ECO [47]	res.	6	1 Hz	≈1 Hz	7–12	138–245[b]	12.5 GB
FIRED* [N20]	res.	1	8 kHz	2 kHz	68	101	3.2 TB
SustDataED [93]	res.	1	12.8 kHz	1/2 Hz	17	10	≈30 GB
BLUED [94]	res.	1	12 kHz	-[c]	43	8	53 GB
REDD [17]	res.	6	15 kHz	≈1/3 Hz	26	5–19[b]	1.7 GB
UK-DALE [75]	res.	5	16 kHz	1/6 Hz	52	38–1580[b]	≈10 TB
BLOND-50 [21]	office	1	50 kHz	6.4 kHz	53	213	15.3 TB
BLOND-250 [21]	office	1	250 kHz	50 kHz	53	50	23.4 TB
WHITED[d] [22]	lab.	-	-	44.1 kHz	110	-	384 MB
PLAID [e][23]	lab.	-	-	30 kHz	235	-	8.2 GB

* The FIRED dataset is a contribution of this thesis and will be introduced in Chapter 6.
[a] small version used in NILMTK [69]
[b] missing data removed
[c] state transitions of each appliance are labeled
[d] includes ten recordings of 5 s length for each individual appliance
[e] includes 1094 recordings in total; length between 2–10 s

The Reference Energy Disaggregation Dataset (REDD) [17] was introduced by Kolter and Johnson in 2011. The authors recorded the whole house electricity consumption of six different homes in the US for 25 to 28 days. High-frequency mains data (15 kHz voltage and current waveforms) of the complete recording duration are, however, only available as compressed files generated with a custom lossy compression. Socket and sub-circuit-level data are only available as unevenly

sampled low-frequency data of approximately 1/3 Hz. Furthermore, the data shows gaps of several days.

The UK Domestic Appliance-Level Electricity (UK-DALE) dataset [75] introduced by Kelly and Knottenbelt in 2015 includes the whole house electricity demand of five homes in the UK. In particular, three of the houses (1, 2, and 5) have been recorded at a sampling rate of 16 kHz. House 1 was recorded for 1629 days, resulting in the longest whole house recording of any found dataset. Nevertheless, individual appliance data was sampled with a comparably low sampling rate of around 1/6 Hz and also contains several gaps.

The Electricity Consumption and Occupancy (ECO) dataset [47] was introduced by Beckel et al. in 2014 and consists of 1 Hz aggregated and individual appliance measurements of six homes in Switzerland. Data dropouts (over 900 days of data are missing in total), a low individual appliance coverage, and especially the low sampling rate makes it difficult to use the dataset for the evaluation of event-based NILM and activity recognition approaches.

The Almanac of Minutely Power dataset (AMPds) [16] was introduced by Makonin et al. in 2013. It features electricity, water, and gas readings at one minute resolution of a residential building in Canada. The authors used an off-the-shelf electricity meter (Powerscout18 by DENT [95]) to record the whole house consumption and the consumption at individual circuit breaker level over a time period of two years. Data of the exact same and one additional home are further available at 1 Hz resolution in the Rainforest Automation Energy (RAE) dataset [92] which was introduced by the same authors in 2018. RAE covers 72 days of electricity data and has been recorded using the same DAQ as AMPds. Still, the data have been sampled at a comparably low sampling frequency (cf. Table 3.1).

In the non-residential domain, Kriechbaumer and Jacobsen proposed the Building-Level Office eNvironment Dataset (BLOND) [21] in 2018. They recorded aggregated and appliance-level data of an office building in Germany over a time period of around 260 days with up to 250 kHz. Their dataset is split into two measurement series. BLOND-50 features 50 kHz aggregated and 6.4 kHz appliance-level data recorded over 213 days, and BLOND-250 features 250 kHz aggregated and 50 kHz appliance-level data recorded over 50 days. For both sets, additional 1 Hz power data has been derived from the voltage and current waveforms. However, downloading the dataset requires to store approximately 40 TB of data. Moreover, the authors have not used their recording system to generate a residential dataset yet.

The Building-Level fUlly-labeled dataset for Electricity Disaggregation (BLUED) [94] introduced by Anderson et al. in 2012 was specifically recorded with event-detection in mind. The authors recorded eight days of voltage and cur-

rent measurements of a home at aggregated level with a resolution of 16-bit and a sampling rate of 12 kHz. Significant power changes (> 30 W) were labeled either manually, using additional sensors, or switchable sockets. While including valuable information about appliance events, the datasets lacks individual appliance electricity recordings.

It is noted that the list of datasets (summarized in Table 3.1) is not trying to be complete. It rather highlights their differences in terms of the number of included homes, covered appliances, data resolution, and recording duration. As, moreover, each dataset is typically stored in a different (file) format, working with them requires to develop a lot of boilerplate-code in order to later use the datasets to benchmark NILM systems.

Summarized, most existing electricity datasets provide comprehensive electricity measurements which allow to identify user behavior over longer time periods. This includes ground truth data, i.e., individual appliance consumption data or - in case of BLUED - the timestamps of appliance events and additional information about the events, allowing to train supervised machine learning algorithms. Furthermore, aggregated appliance recordings of higher frequencies are provided in many datasets which allow to extract appliance features such as higher signal harmonics. However, only the BLOND dataset, which has been recorded in an office environment, features simultaneous high-frequency aggregated and individual appliance recordings. Such data can be used to extract appliance events (as available for BLUED) which can then be utilized as ground truth data to train and evaluate event-based NILM systems. The lack of such a datasets in the residential domain was an additional motivation to record the FIRED dataset which will be presented in Chapter 6.

3.3 Appliance Event Detection

Event-based NILM methods as well as many other use cases for electricity data rely on the analysis of user-induced or self-induced events, i.e., when electrical appliances are being switched on or off, or their mode of operation changes. Events are a subset of all signal *transients*. Transients describe all rapid changes in the power consumption while events only refer to user-relevant changes. User-relevance has, however, not been defined consistently in related works.

In Table 3.2, the numbers of user-relevant events are summarized which have been found in a selection of publicly available electricity datasets. The average of the tabulated values ranges at approximately 275 events per day, i.e., approximately one event every six minutes.

Event detection relies on the concept of the Switch Continuity Principle (SCP) introduced by Hart [96] in 1992 and confirmed to still be valid by Makonin [97] in 2016. The SCP states that at a specific point in time only a single event, i.e., appliance state change, occurs and that, overall, the number of events is small compared to the number of recorded samples in the signal. In other words, events can be assumed to be anomalies in the signal, allowing to utilize a range of known methods for their detection [52].

Table 3.2: Summary of the number of events detected in publicly released electricity datasets.

Dataset	# Events	Timespan	Source of event count
UK-DALE [75]	5440	7 days	Pereira and Nunes [98]
REDD [17]	1944	8 days	Völker et al. [W20]
REDD [17]	1258	7 days	Pereira and Nunes [99]
BLUED [94]	2335	8 days	Anderson et al. [94]
FIRED [N20]	4379	14 days	Völker et al. [J21a]
BLOND-50 [21]	3310	30 days	Kahl et al. [100]
AMPds [16]	651	7 days	Pereira and Nunes [99]
SustDataED [93]	2196	11 days	Pereira et al. [101]

In practice, event detection algorithms span the range from computationally lightweight solutions (e.g., using thresholds between successive power samples [18, 96, 102]) to more complex approaches. Examples of the latter include the application of probabilistic models [C19a, 39, 103] or advanced filters in order to suppress minor fluctuations while emphasizing actual events [104–106]. An overview of these algorithms is provided in Section 3.3.2.

3.3.1 Event Definition

Before discussing related work on various event detection techniques, it should be emphasized again that no uniform definition for an *appliance event* has yet been agreed upon.

Anderson et al. [52] defined an event as a change in power of more than 30 W for a certain amount of time. In turn, Jin et al. [107] stated an event to be a transition between the on and off state of an appliances while Girmay and Camarda [41] informally state an event as an *active region* of an appliance without explicitly stating what the authors mean by *active*. Kahl defined events from a more consumer-centric perspective as "*appliance ON / OFF events that have a causal origin (i.e., from user interaction or physical appliance state changes) [...]. In*

practice, the consumer might be interested in the fridge or washing machine spin cycles. The temporarily increased energy consumption from a laptop during an irregular 5 minute lasting operating system update [...] is only of minor interest to the consumer." [108] As consumer preference is subjective to each individual person, in this work, it is referred to events according to the following more general definition of Wild et al.:

> "*An event is a transition from one steady state to another steady state which definitely differs from the previous one [... or] a so-called active section where the signal is somehow deviating from the previous steady state.*" [109]

The definition allows to include several different events of the same appliance such as state changes of multi-state appliances (e.g., a desktop fan with multiple speed settings). It is not limited to appliance' *on/active* phases such as the definition by e.g., Kahl but also specifically excludes simple signal fluctuations or regions of variable load.

3.3.2 Event Detection Algorithms

According to Anderson et al. [52], event detectors can be split into three categories: (1) Expert Heuristics, (2) Probabilistic Models, and (3) Matched-Filters.

Expert Heuristics:
An expert heuristics describes a detector for which a set of rule-based parameters are fixed by domain experts using prior knowledge of the data.

Hart [96] used a simple predefined power threshold based on the absolute difference of two adjacent samples to mark an event. This concept has been enhanced by Weiss et al. [18], who used a threshold-based setup on the normalized apparent power ($S'_n = (230\,\mathrm{V}/V_{RMS})^2 \cdot S$) which is more resilient against voltage fluctuations than the standard apparent power (cf. Equation 2.10). The signal was further smoothed by the combination of a mean and median filter to remove unwanted signal noise. While a Gaussian-weighted average filter was found to be superior in suppressing noise, it adds quadratic complexity and was, therefore, rejected.

Meehan et al. [102] developed an event detector based on the 1 Hz RMS current signal. They defined two criteria that need to be fulfilled in order to mark an event as such. First, the considered RMS value needs to be greater than the value four seconds earlier plus a threshold value ($I_{RMS}(t_{now}) > I_{RMS}(t_{now} - 4s) + I_{\Delta min}$). Second, the last event must have occurred at least three seconds ago. The last criterion adds the limitation that real events which happen within three seconds will not be identified correctly.

Probabilistic Models:
Probabilistic models use statistical metrics such as mean and variance to estimate the probability that an event has occurred at a certain point in time.

Luo et al. [103] evolved the General Likelihood Ratio (GLR) test originally proposed by Basseville and Nikiforov [110], which calculates a likelihood signal using the logarithm of the ratio between the probability distributions before and after a mean change. Their method is based on sliding windows and requires to optimize four parameters depending on the underlying data, including the length of the windows and a detection threshold. The algorithm has been slightly adapted by other researchers. Berges et al. [111] added a voting scheme to identify the exact point of an event from the likelihood signal. Anderson et al. [52] added a cleaning element and used a slightly different voting method, while Pereira [39] added more sophisticated voting methods.

Jin et al. [107] introduced the Goodness-of-Fit (GOF) measure for the problem of event detection which uses a χ^2 test to identify if data, i.e., a window of a potential event, have originated from a given probability distribution.

Trung et al. [104] used a Cumulative SUM (CUSUM) filter to clean the power signal. Thresholds are then used to detect the starting and end point of a transient.

Wild et al. [109] applied the Kernel Fisher Discriminant Analysis (KFDA) on the first eleven odd current harmonics. Their method uses adaptive thresholds to identify active and steady-state signal regions. These are afterwards post-processed and checked for plausibility by requiring active regions to be at least of the same length as the detection windows and above a certain adaptive threshold level. One drawback of this approach is that it is computational expensive and requires access to the high-frequency current signal.

Azzini et al. [112] proposed the *window with margins* method, which uses a sliding window. Only the window's first and last m samples are used and the difference in mean of these two regions is calculated. If this difference exceeds a certain threshold, a fine search is conducted to find the exact sample of the event. However, the fine search is not detailed any further. A second, larger window with margin is afterwards utilized to verify the event. To reduce the computational complexity of their method, the authors proposed to replace the sliding window by a simple threshold test on the signal's derivative.

Further methods based on the change of mean scheme mainly differ by the used data filtering strategy or the used features. Berriri et al. [113] employed the effective residuals, while Cox et al. [106] made use of the spectral envelope of the first and third harmonic of the voltage signal. De Baets et al. [105] used spectral components which have been smoothed by an inverse Hann window in the Cepstral domain.

Matched-Filters:
Matched-filter approaches correlate a mask, i.e., a known signal of the appliance event, which has been recorded in advance, with the examined signal to detect a correlation with the mask and, thus, the presence of an event.

In [114], Leeb et al. proposed an event detector which matches known start-up current events to the aggregated signal. This is achieved by applying two hardware transversal filters which identify shape and amplitude matches. A certain threshold tolerance is set, since a perfect match is unlikely in noisy environments.

Zheng et al. [115] proposed an event detector based on Density-Based Spatial Clustering of Applications with Noise (DBSCAN). The clustering algorithm is applied to a moving window. If an event is present within the window, two found clusters reveal two steady states. This indicates that a transition (aka event) between these steady states must be present within the window. The values in the transient intervals are regarded as outliers by the algorithm and do not form separate clusters. In addition to the timestamp of each event, their method, therewith, also reveals the average steady-states values before and after the event. They further state that their detector is invariant to a certain amount of noise, longer transient intervals, and can detect small power changes as well.

Other Methods:
More recently, Jorde et al. [116] proposed an unsupervised event detector based on the CUSUM which serves as the input of a bidirectional recurrent denoising autoencoder. An autoencoder is an Artificial Neural Network (ANN) comprised of a set of encoder and decoder layers. The goal of the autoencoder is to reproduce the input data as close as possible while transferring the data through a layer with significantly less neurons, preserving only the most relevant features of the data. The approach by Jorde et al. outperformed other state-of-the-art event detectors based on GOF and DBSCAN on the BLUED and BLOND datasets.

Kahl et al. [117] proposed a multivariate event detector, which uses explicitly labeled events and non-events as well as implicit non-events. The latter are selected randomly from data between known events. In multiple adaptive training runs, false positive events are continuously added to the non-event class, making the class more heterogeneous each iteration.

For a recent and comprehensive comparison of multiple event detectors, the reader is referred to Houidi et al. [118]. The authors found that detectors which are based on either the CUSUM or the GLR are amongst the best performing algorithms.

Since appliance event detection methods rely on the validity of the SCP, it seems obvious that higher sampling rates are beneficial for the performance of most event detectors. If the sampling rate is low (e.g., one sample every 6 s such as in the

individual appliance data of the UK-DALE dataset [75]), it cannot be assumed that the SCP holds as multiple appliances state changes may occur between two adjacent samples. The higher the sampling rate of the ground truth and the data itself, the more samples can be used to detect and classify events when they happen in close temporal proximity.

The BLOND dataset with its very high sampling rate of 250 kHz allows to even identify the individual spikes of different SMPS-powered appliances within a single mains period [119]. This, however, comes with the drawback of a huge amount of data which needs to be processed and stored (see Table 7.2).

3.4 Appliance Classification

Appliance classification, sometimes referred to as appliance identification or event classification (cf. Figure 2.2), follows the pattern recognition paradigm and can be approached using machine learning methods. Features, which are typically handcrafted by domain experts, are extracted from each event and are fed into a classifier which outputs more details about an event (e.g., a specific appliance turning on). As the inference step is only applied to each new event and events are typically rare (cf. SCP [96]), event-based NILM systems are typically computationally less expensive compared to event-less approaches which perform the inference step for each new sample.

3.4.1 Features

Over the years, several handcrafted features have been introduced by various researchers for the task of appliance classification.

G. W. Hart's first prototype [96] relied on low-frequency active (P) and reactive (Q) power measurements. If a change of active power is observed, the relative changes for P and Q are mapped onto a PQ-plane and matched to previously extracted tuples of known appliances. Even if this simple prototype already performed quite well, Hart noted that further features such as harmonics need to be considered for appliances that are mapped to the same cluster in the PQ-plane. Furthermore, the approach requires an intrusive and preceding training phase in which all appliances need to be switched on and off multiple times.

In 2000, Chan et al. [120] suggested to use a discrete wavelet transform to capture harmonic signal components. They found that typical non-linear loads (they used a PC, a fluorescent light, and a dimmer) can be distinguished quite well by using

a five-level wavelet decomposition. Furthermore, their method is robust to noisy environments as all level-1 wavelet coefficients (which represent higher frequencies) are rejected.

To distinguish appliances which show similar characteristics for traditional power metrics such as active power or frequency components, Lam et al. [121] introduced a two-dimensional form of a load signature (i.e., a unique appliance fingerprint), called the V-I Trajectory (VIT). The VIT incorporates the load signature of voltage and current into a two-dimensional shape feature. This shape was observed to be unique for different appliances even if they are of the same or similar type (e.g., two televisions or a television and a monitor). This high discriminative power has likewise been proven by other researchers such as [117, 122, 123].

Yang et al. [124] suggested the Total Harmonic Distortion (THD) which is the ratio of the accumulated sum of all harmonic frequency components f_i related to the fundamental frequency f_0. They further stated that a steady-state analysis is not sufficient to detect concurrently running appliances and suggested to further research the combination of steady-state and transient features (i.e., features derived from the steady-state and the transient portion of the event) to improve recognition accuracy.

Gupta et al. [125] followed a different approach by analyzing Electromagnetic Interference (EMI) in the voltage signal up to 500 kHz. EMI are generated by interfering external sources (the appliances in this context) caused by electromagnetic induction. Especially appliances powered by SMPS can be distinguished well using their approach. Since only the voltage signal is analyzed, linear loads which do not induce EMI are hard or impossible to recognize solely using EMI analysis. Furthermore, EMI induced by appliances which are not located in the home (e.g., appliances of a neighbor) are also visible and might adulterate the system's performance.

Gao et al. [126] tested several classifiers with several features including the VIT converted into a binary image. This laid the foundation for the successful application of Convolutional Neural Network (CNN) based image classification algorithms on such binary VIT images by e.g De Baets et al. [127]. The results by Gao et al., however, also highlight, that a combination of several features from different domains (such as statistics, electrical engineering, or speech recognition) will potentially lead to significant performance gains.

Surveys by Liang et al. [128], Kahl et al. [117], or Sadeghianpourhamami et al. [129] offer a comprehensive listing of features used for appliance classification. Kahl et al. [117] evaluated 36 features in a standalone feature analysis as well as their combination using a *feature forward selection* technique. This technique constructs the best performing feature set iteratively by including the next feature

into the existing set that shows the best performance amongst all possible combinations until the performance cannot be increased anymore. The authors found that across all datasets used, the phase angle difference between voltage and current ($cos\,\Phi$) was the best scalar feature ($F_1 = 0.49$) and Current Over Time (COT) achieved the best multi-dimensional feature performance ($F_1 = 0.8$). The best feature combination differed depending on the dataset and could not be generally determined.

Summarized, standard electrical measurements such as active power, features which stem from feature engineering such as VIT, and frequency domain features such as signal harmonics are promising candidates for NILM systems. While EMI has shown promising results, it requires expensive data acquisition hardware and huge processing capabilities which cannot be provided by existing smart meter infrastructure.

3.4.2 Algorithms

Different classification algorithms have been evaluated for the task of appliance classification such as Random Forests (RFs) [129–131], Support Vector Machines (SVMs) [19], k-Nearest Neighbour (kNN) [19, 131, 132], and more recently Artificial Neural Networks (ANNs) [127, 130, 133]. Several of them have been added as benchmark algorithms to the NILMTK framework [69]. NILMTK is a test-bed for NILM and allows to compare and evaluate low-frequency NILM methods on different datasets. However, support for high-frequency methods has not been established yet.

Hart's initial prototype [134] uses clustering techniques to match pairs of on and off events based on the measured active and reactive power. During supervised training, a user has to provide the appliance names for the detected clusters. Hart's method has been included as a benchmark algorithm in the NILMTK framework [69].

Kolter and Jaakkola [44] explored the use of Factorial Hidden Markov Models (FHMMs) for NILM. They adapted a standard additive FHMM by using a difference signal, by including a mixture component to account for non-modeled observations, and by allowing only a single hidden state to change at each time step. Their algorithm showed promising results on the REDD dataset with precision/recall scores of up to 87.2 %/60.3 %. Zhong et al. [135] adapted the approach and added explicit domain knowledge into the FHMM.

Kramer et al. [136] compared plain kNN, SVM, and Decision Tree (DT) classifiers to bagging ensembles, such as RF or the combination of kNN and SVM. Ensembles

combine the inferences of multiple learners via averaging. The authors found that such ensembles tend to perform slightly better, especially as the size of the training set increases.

Kelly and Knottenbelt [70] introduced one of the first works in which neural networks are applied to NILM in 2015. They compared three architectures. One based on Recurrent Neural Network (RNN), a denoising autoencoder, and a network which lays rectangles over appliance usage periods. The height of the rectangle corresponds to the estimated average power consumption of the appliance. They found that the denoising autoencoder and the *rectangles* architectures perform well, especially on unseen houses with up to 99 % of correctly assigned energy (cf. Section 2.5).

Jorde et al. [133] proposed the first approach that applied deep neural networks to raw high-frequency measurements. In particular, the authors used data from the WHITED [22] (44.1 kHz) and PLAID [23] (30 kHz) datasets (cf. Table 3.1). Their method uses data augmentation techniques based on phase shift and half-phase flip to increase the training set by a factor of ten. Several binary deep CNN are trained for each appliance and are combined in a *one-vs-rest* strategy. A perfect F_1-score was achieved on WHITED and around 69 % on PLAID.

Huber et al. [30] and Kahl et al. [117] surveyed several algorithms for appliance classification. Huber et al. [30] focused on Deep Neural Networks (DNNs) and identified higher sampling rates, the use of larger receptive fields, and an ensemble of input features, amongst others, as promising techniques to improve the performance of such networks. Kahl et al. [117] directed their focus on standard machine learning algorithms and identified that kNN performs quite well for the task of appliance classification despite its comparable low computational complexity.

Summarized, FHMM and neural network based methods are better suited for low-frequency and event-less NILM methods. While ANN-based methods also find their applications for high-frequency methods in literature, classical machine learning algorithms such as kNN or SVM are typically used for high-frequency and event-based NILM methods. It is further noted that the training of ANNs constitutes a large burden for resource constrained embedded systems such as smart meters. Depending on the system's restrictions, a computationally lightweight algorithm such as kNN may be better suited.

3.5 Data Labeling

The development and evaluation of event-based NILM systems requires datasets with precise ground truth data, i.e., event labels in the individual power consump-

tion of each appliance. Some electricity datasets were specifically recorded with event-detection in mind and deliver appliance events as part of their ground truth data (e.g., switching events in the BLUED dataset [94]). Adding such ground truth information after the recording remains challenging, as it requires human inspection of the data and expert knowledge. For this purpose, researchers have developed semi-automatic labeling approaches, such as [99] proposed by Pereira and Nunes. Their system uses the Log-Likelihood Ratio (LLR) test to identify events in the signal which are later refined by an expert rater. Furthermore, crowd sourcing and gamification techniques were explored by Cao et al. [137] to enable collaborative ground truth labeling and to apply the *wisdom of the crowd*. Such systems have already shown their potential in similar domains such as image classification (e.g., CAPTCHAs [138]).

Tools to label electricity data have been published e.g., by Pereira et al. [101] and Huchtkoetter et al. [139]. Although these tools exist, the inspection and labeling of public electricity datasets or newly recorded data still remains cumbersome. Datasets have to be downloaded from the Internet and need to be stored locally first. The sheer size of some datasets (cf. Table 7.2) requires careful preparation. Since no file format has been established, each dataset requires specific code to load data into memory before even an initial review of the data is possible.

Adding labels to those datasets is a challenging and time-consuming task as several appliances change their state regularly resulting in hundreds of events per day. Pereira et al. [101] evaluated their labeling tool and were able to find 94 % of all events automatically. The main shortcomings of their approach are that only appliance events with a corresponding power change of at least 30 W can be labeled and that no textual descriptions can be added to events.

4 Electricity Data Acquisition Framework

This chapter lays the foundation for a hardware and software framework to acquire versatile high-frequency electricity datasets. The employed techniques make use of the contributions of [J21a], [C19a], [C19b],[PA18], and [BC19] and mainly include:

- Definition of a set of requirements for electricity datasets (see Section 4.1) [J21a].
- Design and development of a modular Data Acquisition Systems (DAQs) to collect the aggregated electricity consumption of a home (see Section 4.2) [J21a, C19b].
- Design and development of a modular DAQ for high-frequency measurements of the electricity consumption of individual appliances (see Section 4.3) [J21a, PA18].

4.1 Requirements and Design Goals

Comparing different NILM algorithms on existing datasets remains a challenging task, since not all algorithms can be applied to the same datasets due to specific input data requirements of each algorithm. To test the performance of event-less methods, and if these methods are supervised, also for their training, individual appliance consumption data are needed. If event-based methods are trained and evaluated, additional ground truth data of appliance events are required to evaluate the event detection step (cf. Figure 2.2) separately. Moreover, event-based methods typically require data acquired at higher sampling rates to reliably detect events and still distinguish between two different events that are close in time. In addition, high-frequency methods require data with a sampling rate greater than twice the maximum frequency being analyzed (according to Equation 2.5). BLUED [94] was specifically recorded with event-detection in mind as all significant appliance events were labeled manually. However, no individual appliance electricity measurements are available in this dataset, which in turn are needed whenever the disaggregation

step (cf. Figure 2.2) is supposed to be evaluated. Further shortcomings of existing datasets were identified and are summarized in the following.

(1) Larger time periods where no data or only parts of the data are available. For instance, 13 days are missing across the complete recording duration in house 1 of the REDD dataset. In the ECO dataset, over 900 days of data are missing (summed over all homes and meters).

(2) Comparatively low sampling rate of the appliance-level data (e.g., 1 Hz for ECO, 1/3 Hz for REDD, 1/6 Hz for UK-DALE, and 1/60 Hz for AMPds), or no appliance-level data at all (BLUED).

(3) Appliance events have not been labeled, i.e., timestamps and textual description of events are missing (e.g., for REDD, ECO, UK-DALE, BLOND).

(4) Unknown number and type of appliances measured at the aggregated point (e.g., ECO, UK-DALE, BLUED).

(5) No standard procedure to load the data or explore the dataset.

Therefore, a set of challenges have been defined that should be handled when recording datasets used to evaluate a variety of load monitoring or other electricity-related algorithms in the residential domain. The challenges are summarized below.

C1 **Simultaneous recordings** of a home's aggregated electricity consumption and the consumption of the individual appliances are necessary. The individual data can be used to validate the appliance estimates of NILM algorithms. Furthermore, it can be explored how semi-supervised hybrid NILM algorithms such as [PA18] can benefit from individual appliance data.

C2 **High sampling rates** of the aggregated and individual appliance data are required. This allows to extract and utilize high-frequency features from the individual waveforms which might further improve traditional NILM algorithms (cf. Section 8.2). Kriechbaumer and Jacobsen [21] focused on recording a dataset with a high sampling rate (of up to 250 kHz for BLOND-250). However, using this dataset requires to download approximately 75 GB of data per day. To avoid sacrificing usability, a trade-off between high sampling rates and data size needs to be examined.

C3 **Continuous data recording** over several days is crucial to understand and study different consumption and, thus, user behavior depending on the *time-of-day* or *day-of-week*.

C4 **Dataset labels** are needed to evaluate event-based NILM and event detection algorithms. These labels should consist of a timestamp describing when the event occurred, the appliance responsible for the event, and a textual

description of the event. BLUED also contains a timestamp and the corresponding appliance for each event in the power signal ($> 30\,\mathrm{W}$) but lacks additional information about the event (such as "fridge door opened").

C5 **High temporal accuracy** of the data and its labels are required. Labels should always reflect the associated change in the aggregated and individual measurements. This requires that the measurement data and the labels are time-synchronized and do not drift apart.

C6 **Safe usage** while using all components. Home appliances are typically powered by higher voltages (e.g., $230\,\mathrm{V_{RMS}}$ in Europe) which need to be measured by the recording hardware. Thus, the recording system itself requires several safety measures.

C7 **Usability** is one of the most underrated factors of a system. Researchers should be able to use systems or publicly available dataset in a quick and easy way.

Table 4.1 summarizes the above stated challenges that have been addressed by existing NILM datasets. It is noted that although the challenges have been specified for a recording framework, they can be directly related to the corresponding datasets with the exception of challenge C6. However, it is considered that each recording hardware used to record a dataset has been designed with safety in mind.

Table 4.1: A comparison of which challenge is met by different NILM datasets. recorded at higher frequencies ($> 1000\,\mathrm{Hz}$) are considered. The recording hardware of each dataset is assumed to meet challenge C6.

Dataset	Domain	C1	C2	C3	C4	C5	C7
WHITED [22]	lab.	-	-	-	(✓)[a]	✓	-
PLAID [23]	lab.	-	-	-	(✓)[a]	✓	✓
BLUED [94]	res.	-	-	-	(✓)[a]	✓	-
REDD [17]	res.	✓	-	-	-	-	-
UK-DALE [75]	res.	✓	-	✓	-	-	-
BLOND-50 [21]	office	✓	✓	✓	-	✓	-
BLOND-250 [21]	office	✓	✓	✓	-	✓	-

[a] the datasets include only the name of the appliance causing the event; no additional data are provided

WHITED, PLAID, and BLUED only partially satisfy challenge C4, since the corresponding event information only includes the appliance causing the event. However, the specific information about what happened during the event, such as a

channel change of a television, might be of additional interest. Table 4.1 further highlights that there is still no *one-fits-all* solution. The BLOND dataset has the highest potential to satisfy all challenges. Additional event information might further be extractable from the individual high-frequency recording. Furthermore, usability can be improved by providing code that downloads the data on demand. However, the dataset is recorded in the office domain and, therefore, may not be directly transferable to the residential sector. Based on the stated challenges, a framework to record and label NILM datasets is developed. The overall flow of this framework is shown in Figure 4.1.

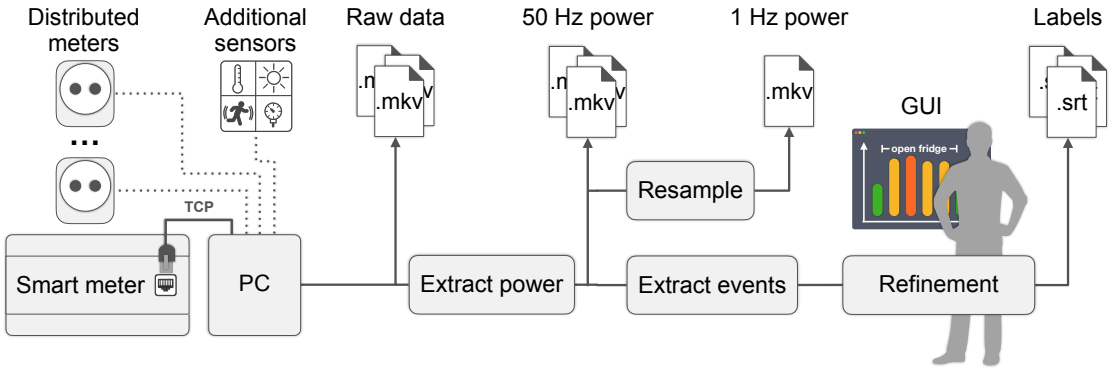

Figure 4.1: Flow of the presented framework: A smart meter and multiple distributed meters sample electricity data gathered by a recording PC. Additional data are recorded using environmental sensors. The raw data are stored and additional measures are derived. Event labels are automatically added, refined by a human, and stored into files.

It consists of an aggregated electricity meter (depicted as the smart meter in Figure 4.1) that records high-frequency voltage and current waveforms at aggregated level. Furthermore, the framework includes multiple distributed meters which record voltage and current waveforms of individual appliances. Additional sensors can be added to measure other quantities (e.g., temperature or movement). The current and voltage waveforms as well as the sensor data are collected by a recording PC and are stored in multimedia containers. Other electricity-related quantities such as active and reactive power are derived from the raw voltage and current waveforms. These power data are stored with different (smart meter like) sampling rates and can be used to semi-automatically generate data labels. A post-processing step extracts events and assigns labels to these events. Both, event positions and labels are refined by a human using a GUI, resulting in a final set of label files. The hardware and software components to acquire and congregate electricity data and other measurements are detailed in the remainder of this chapter. Event extraction and labeling is explained separately in Chapter 5.

4.2 Smart Meter

Aggregated-level data can be gathered using a custom-built measurement system which in the following will be termed as the *SmartMeter*. The SmartMeter is a cyber-physical system consisting of several hardware and software components tightly integrated into a single enclosure, as shown in Figure 4.2.

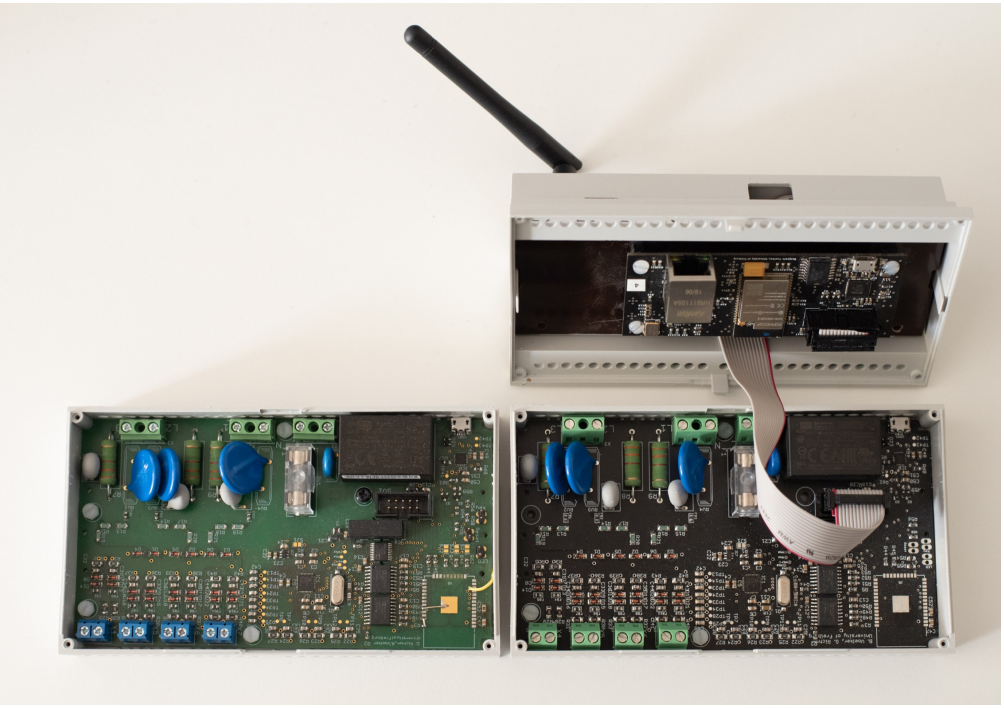

Figure 4.2: Two versions of the SmartMeter hardware. Initial (left) and the modular second iteration (right) with an expansion board for ethernet and USB serial connection.

The system offers access to different communication channels either via a versatile wireless channel or via different robust wired connections (USB and Ethernet). A dedicated electricity monitoring chip is integrated to capture different electricity data of multiple grid lines inside a home‘s power grid.

A high-level system overview is illustrated in Figure 4.3. A galvanic isolation layer, according to DIN EN 60664 [140], divides the SmartMeter into two separate assemblies, i.e., Printed Circuit Boards (PCBs). This separates higher voltage levels present at the analog frontend from the microcontroller and communication interfaces and, thus, also from external systems attached via a cable connection.

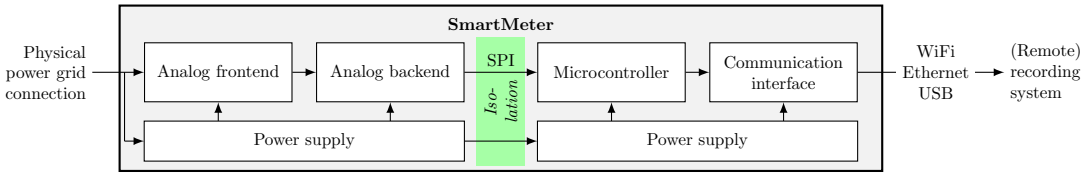

Figure 4.3: System architecture and data flow of the SmartMeter. The analog frontend converts the physical quantities into measurable voltage levels; the backend converts these into the digital domain. The values are read out by the microcontroller via a galvanically isolated SPI interface. The values are send via a communication interface to an external data sink.

4.2.1 Analog Signal Processing

The central component of all digital measurement systems which sense analog signals, such as voltage levels and current flows, is an ADC. To measure voltage and current of all supply legs of a home (typically three legs in Germany), six ADCs are required if the signals ought to be sampled simultaneously. Several companies such as *Analog Devices* or *STMicroelectronics* offer electricity monitoring ADCs dedicated to analyze electrical power grids. These Integrated Circuits (ICs) further offer an additional Digital Signal Processor (DSP) to calculate other electricity-related metrics such as active power or electrical energy. By considering quantities such as the number of analog channels, SNR, resolution, and maximum sampling frequency, it was decided to use the *ADE9000* from *Analog Devices* [141] for the SmartMeter. The ADC can handle seven input signals at a resolution of 24-bit and a SNR of 96 dB. It further features an internal DSP and can operate up to a maximum sampling frequency of 32 kHz (cf. C2). To avoid aliasing artifacts, all analog signals are band-limited by a first order RC low-pass filter with a cutoff frequency (attenuation of -3 dB and -20 dB/dec) of $f_c = 7.5$ kHz. Furthermore, analog signal lines are routed symmetrically to prevent that interference only effects a single signal line.

Voltage Sensing:
The circuitry to sense a single grid line's voltage is shown in Figure 4.4. The SmartMeter utilizes three of these circuits, to measure the voltage of all three grid lines. A voltage divider (consisting of R_{HSR}, R_{PTC}, R_{1-3}, and R_{sens1}) with a ratio of around 1:995 reduces the grid voltage level of approximately $\pm 320\,\mathrm{V_{peak}}$ to approximately $\pm 322\,\mathrm{mV_{peak}}$. To protect the circuit from incorrect wiring or transients up to 6 kV (according to category IV of norm *DIN VDE 0100-443* [142]), a varistor (RV) and a PTC thermistor (R_{PTC}) are added. Further diodes (D_1, D_2) prevent that voltage levels of more than 1 V can be applied to the ADC (cf. C6).

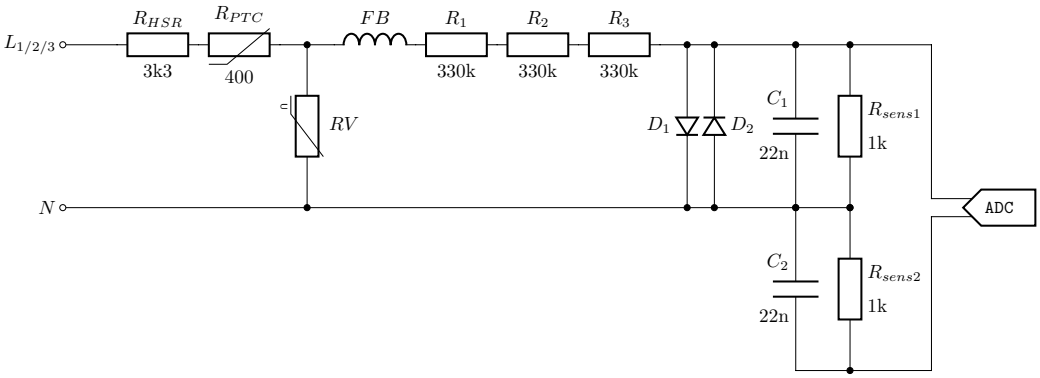

Figure 4.4: Circuitry to sense the mains voltage using a voltage divider.

Current Sensing:
The circuitry to sense the current flowing through a single grid line is shown in Figure 4.5. The current is scaled down using a non-intrusive burden-less current transformer (*SCT-013-000* from *YHDC* [143]). The CT has a transfer ratio of 1:2000 and can be easily attached to the existing electrical wiring infrastructure due to its split core. R_{sens1} and R_{sens2} act as the CT's burden resistor. A burden is a measuring shunt over which a voltage can be measured which is proportionally to the current flowing through it. A larger burden resistor allows for a larger current resolution as the same current results in a larger voltage level (cf. Ohm's law as $U = R \cdot I$). However, a CT should ideally be loaded with as little resistance as possible, since otherwise the linearity of the CT suffers due to core saturation. As a trade-off, 15.74 Ω ($R_{sens1} + R_{sens2}$) are used resulting in a maximum measurable current of 127 A using an ADC amplification of 1. Again, diodes (D_{1-4}) are used for over-voltage protection of the ADC (cf. C6).

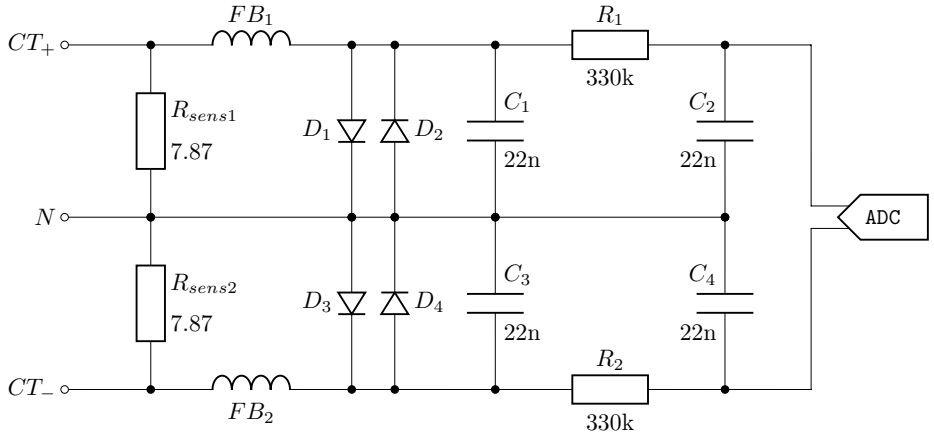

Figure 4.5: Circuitry to sense the mains current using a CT.

4.2.2 Galvanic Isolation

The microcontroller and communication interface was further isolated from the high-voltage analog parts, allowing to safely interface via the wired communication channels of the SmartMeter (cf. C6). An SPI-isolator (*ADUM3150ARSZ* by *Analog Devices*) and two IO-isolators (*ADUM3401CRWZ* by *Analog Devices*) are used to separate the communication interface and all control lines between the ADC and the microcontroller. The power supplies of the ADC and the microcontroller are further isolated using an additional DC-DC-converter (*MEF1S0303SP3C* by *Murata Power Solutions*).

4.2.3 Digital Sampling and Communication

The sampled data are retrieved by an *ESP32* microcontroller (by *Espressif Systems* [144]). The ESP32 features two 32-bit cores running at 240 MHz, 520 kB SRAM, and 4 MB flash memory. The chip further supports the WiFi standards 802.11 b/g/n and Bluetooth 4.2. The particular version used includes 8 MB external flash memory which is used for data buffering in case of communication dropouts (cf. C3). 8 MB can buffer an equivalent of approximately 41 s of data at a sampling rate of 8 kHz. Furthermore, the board contains a Real Time Clock (RTC) (DS3231 by *Maxim Integrated* [145]) for time keeping and to sync the internal clock of the ADC. Since WiFi reception in smart meter environments (i.e., in a fuse box that is often located in the basement of a building) is typically low, support for external antennas was added via the integration of an *SMA* connector. To not solely rely on wireless connectivity, an Ethernet connection was added to the board via an ordinary *RJ-45* connector paired with the *LAN8720* PHY (by *Microchip Technology*). In Germany, each newly constructed building has to include Ethernet connectivity inside the fuse box according to *VDE-AR-N 4100* [146]. This standard was developed specifically to provide future smart meter installations a connection to the Internet (either directly via the meter or via a gateway). In addition to Ethernet, an *FTDI232H* [147] IC adds a USB serial interface that supports up to 12 MBaud, allowing a direct connection of a dedicated processing unit next to the smart meter.

4.2.4 Firmware

Upon request, the microcontroller reads the ADC values in a continuous loop. These values are calibrated using corresponding calibration parameters for each of the six channels which are stored in the microcontroller's non-volatile memory.

Furthermore, the raw 24-bit values are converted to 32-bit float values representing the actual voltage and current measurements in *Volt* and *Milliampere*, respectively. While this adds an overhead of 25 % to the data, it allows to directly use the data without the need for scaling, conversion, or calibration at the data sink, further enhancing usability (cf. C7). Once in sampling mode, the ADE9000 releases a new interrupt when new ADC samples are available. This in turn triggers the ESP32 to retrieve these values over the galvanically isolated SPI interface. The DS3231 RTC further triggers an interrupt each second. If the number of samples gathered ($n_{current}$) does not match to the target sampling rate (f_{target}) as $n_{target} = (t_{now} - t_{start}) \cdot f_{target}$, samples are left out or are repeated such that the data are as close to the user specified sampling rate as possible. This two-stage process adds only a minor jitter to the data and allows to maintain the sampling rate over long time periods (cf. C3 and C5). The ADC runs at a fixed sampling frequency of either 8 or 32 kHz. Other sampling rates are not supported by default. However, naive support for integer dividers of 32 kHz has been added via sample averaging to provide more flexibility in terms of sampling frequency (cf. C7). If e.g., a sink wants to receive data at a rate of 1 kHz, the ADC is configured to sample at a rate of 8 kHz and a new measurement is created by averaging each eight consecutive ADC samples.

A continuous data stream can be requested by a sink either over *USB*, *TCP*, or *UDP* connection. Different flow control mechanisms allow to minimize the possibility of data loss. Furthermore, data integrity is maintained by successive packet numbers included in each sent data chunk.

4.2.5 Modular Expandability

The SmartMeter incorporates several Internet of Things (IoT) design methodologies to ensure later expandability without loosing functional features. The initial SmartMeter version as well as the second iteration (see Figure 4.2) feature an expansion header that provides direct access to the SPI pins of the incorporated ADC while still maintaining galvanic isolation for safety (cf. C6). This allows not only to test the correct functioning of the ADC after assembly but also to provide direct access to the ADC without the need to relay the data over the integrated microcontroller. An example expansion board was later developed (see top right of Figure 4.2) to add certain of the aforementioned features, such as more storage for buffering, Ethernet connectivity, faster USB serial connection, and time synchronization via an RTC. If the intended use case does not require these features, the modular PCB design, especially of the second design iteration, allows to build a version of the SmartMeter which only includes a single PCB, just like the initial prototype. This can further reduce the system's complexity, cost, and size.

4.2.6 Evaluation

The ADE9000 ADC is used because it is a versatile electricity monitoring IC that can sample up to seven input channels at a maximum sampling rate of 32 kHz. It, therewith, meets challenge C2. If the current amplification factor is set to the maximum, the full-scale current represents approximately 31.77 A_{peak}. At a maximum voltage amplification factor, the full-scale voltage represents 497.35 V_{peak}. With the ADC's vertical resolution of 24-bit, it can theoretically measure current in steps of 3.79 µA and voltage in steps of 59.29 µV. This results in a minimal detectable power change of 12.25×10^{-10} W. If the ADC's SNR is taken into account, the maximum theoretical resolution N of the ADC can be calculated using Equation 4.1 (cf. Kester [148]).

$$N = \frac{SNR - 1.76}{6.02} \tag{4.1}$$

A SNR of 96 dB allows a maximum resolution of $N = 16$-bit. Therewith, current can be measured in steps of 1.94 mA, voltage in steps of 30.36 mV, and power in steps of 58.9 µW.

To verify that the hardware works as intended and to show the high sensitivity, test measurements were performed with different electrical loads. An excerpt of such a test recording is shown in Figure 4.6. The SmartMeter is able to record an appliance power consumption over time (first plot in Figure 4.6, cf. C3) as well as to capture transients and certain appliance characteristics in the high-frequency voltage and current waveforms (second and third plot in Figure 4.6, cf. C2). Thereby, the system is capable of measuring comparably small electrical consumers e.g., battery chargers, without loosing the ability to measure typical high-power household appliances such as kettles. The SmartMeter was specifically designed for long-term continuous measurements (cf. C3). The system was successfully used to collect over 100 days of data without major data loss (see Chapter 6).

The hardware is encapsulated in a fireproof DIN housing. This allows the system to be installed at a DIN rail inside the fuse box. Three SmartMeters have been installed in three different locations (in an apartment, a house, and a university building) for a time period of more then one year and are still installed at the time of writing. So far, no issues such as hazards or power failures, associated with the meter installations, could be identified. The meters continue to operate for their intended applications. At the time of writing, the meter installed at the university building continues to produce up to 12 GB of electricity data per day.

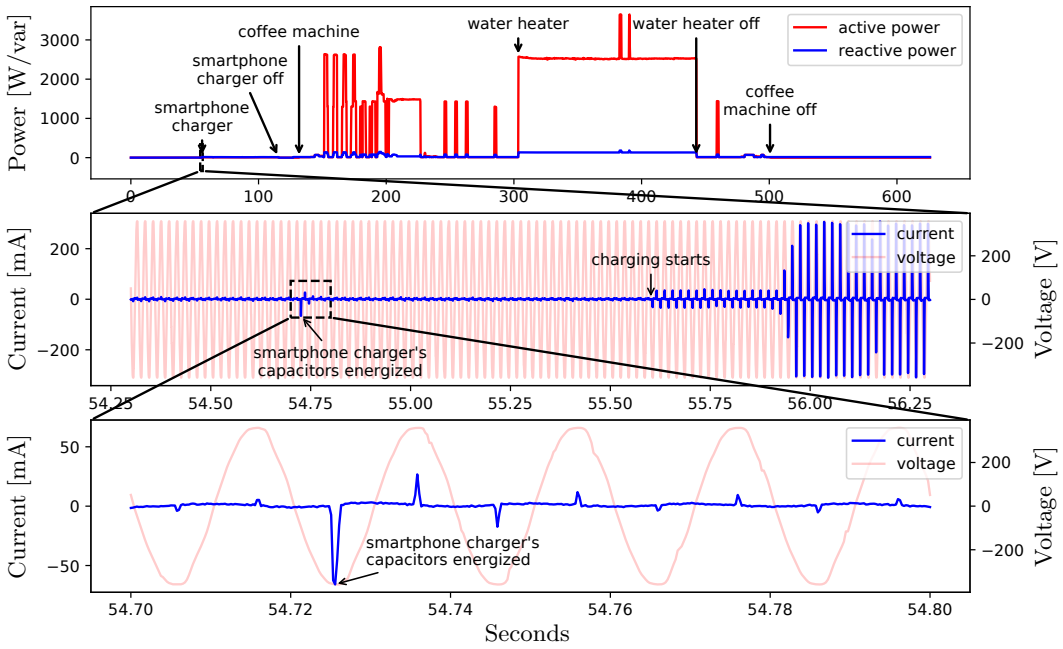

Figure 4.6: Excerpt from a recording of multiple electrical loads (smartphone charger with 5 W, coffee machine with 2500 W, kettle with 2200 W) using the SmartMeter hardware.

4.3 Individual Appliance Meters

Current and voltage waveforms of individual appliances can be recorded using custom-built smart plugs which will henceforth be referred to as *PowerMeters*. A PowerMeter shares the same design principles and also parts of the overall architecture as a SmartMeter (see Figure 4.7) with slightly more focus on wireless connectivity.

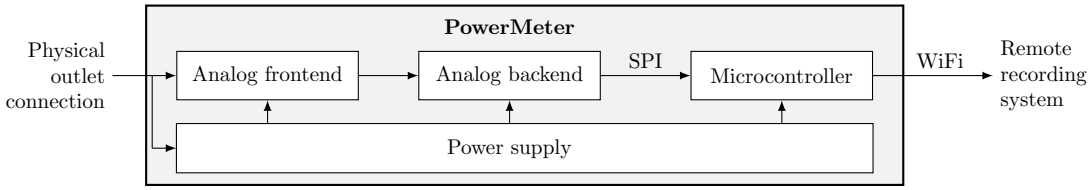

Figure 4.7: System architecture and data flow of the PowerMeter. The analog frontend converts the measurands into measurable voltage levels; the backend converts these into the digital domain. The values are sampled by a microcontroller via SPI and wirelessly sent to a data sink.

To simultaneously record the individual consumption of several appliances, multiple PowerMeters need to be distributed across a home. As individual recordings from a PowerMeter may later be linked to aggregated recordings, precise time synchronization is performed utilizing RTCs and the Network Time Protocol (NTP). For persistent storage, the data needs to be sent to an external data sink. The similarity of the PowerMeter and the SmartMeter can also be seen when comparing their overall system architecture (see Figure 4.7 for the architecture of the PowerMeter and Figure 4.3 for the architecture of the SmartMeter). As the PowerMeter is primed for wireless data transmission and encapsulated in a fireproof housing, no galvanic isolation as for the SmartMeter is required. The PowerMeter hardware is shown in Figure 4.8.

Figure 4.8: Two versions of the PowerMeter hardware. Initial plain PCB (left) with a serial connector and second iteration (right) with a modular expansion header and primed for continuous wireless data streaming.

4.3.1 Analog Signal Processing

A PowerMeter samples voltage and current waveforms at a single grid line (i.e., of a single appliance or a subset of appliances connected to the same outlet). For this purpose the *STPM32* power monitoring IC from *STMicroelectronics* [149] is

incorporated. The chip can handle two analog input signals using two second-order 24-bit sigma-delta ADCs with a maximum sampling frequency of 7.875 kHz (cf. C2). It further includes a DSP to calculate electricity-related quantities such as active power or electrical energy.

Voltage Sensing:
The voltage level of the socket to which the PowerMeter is connected to is scaled down by a factor of approximately 1:1723 using a voltage divider. This results in an ADC input voltage range from $-189\,\text{mV}$ to $189\,\text{mV}$ for the corresponding mains voltage from $-325.27\,\text{V}_{\text{peak}}$ to $-325.27\,\text{V}_{\text{peak}}$ ($\equiv 230\,\text{V}_{\text{RMS}}$), respectively.

Current Sensing:
While non-intrusive split-core current transformers are used to measure current with the SmartMeter, the PowerMeter uses a shunt resistor instead. This is a more accurate but also more intrusive solution, as the current path has to be rerouted over the shunt and the PCB has to be designed to withstand the amount of current. While this would be inappropriate for the SmartMeter, it is not much of a problem for the PowerMeter, as the amount of current is smaller and the plug has to be connected between the outlet and the appliance anyway. The incorporated shunt has a resistance of $3\,\text{m}\Omega$ and is connected in series to the load (i.e., appliance). The voltage drop across the shunt resistor has a current sensitivity of $3\,\text{mV/A}$. Both, the current and voltage paths are designed based on the STPM32 evaluation board and use first order *RC* low-pass filters to suppress higher frequencies.

4.3.2 Controlling Connected Appliances

In order to switch connected appliance *on* and *off*, each PowerMeter includes a 16 A bi-stable relay (*RT314F03* by *TE connectivity* [150]). Bi-stable relays are more expensive but require only power while changing their state. Requiring little to no steady-state power allows to reduce the system's overall power consumption. In addition to the obvious benefit of saving electrical energy, the percentage of energy consumed by the recording framework compared to the actually measured appliances is reduced. Furthermore, it allows to use a more compact 2 W power supply inside the PowerMeter (*IRM-02-3.3* from *Mean Well* [151]), decreasing its overall dimensions. Due to the relay's high switching current of around 200 mA, additional bipolar transistors prevent a damage to the microcontroller pins.

4.3.3 Digital Sampling and Communication

The sampled data are retrieved by the same *ESP32* microcontroller also found in the SmartMeter. Likewise, PowerMeters also include a DS3231 to synchronize

the sampling process (cf. C5) and 4 MB external flash memory for data buffering. The latter allows to be resilient against approximately 250 s network dropouts at a sampling rate of 2 kHz (cf. C3). While the initial PowerMeter prototype includes a serial interface, it was removed in the second iteration to save costs and physical space, since all interfaces would require additional galvanic isolation (cf. C6). Instead, an extension header was added. This either allows to add back such a serial interface using a separate PCB or to add additional sensors e.g., to measure room temperature or humidity levels (see Section 4.3.5). Since a PowerMeter, therefore, does not include a direct physical interface, data has to be retrieved wirelessly. The intended use case of the PowerMeters in which data are gathered in highly-distributed setups would further not allow to route dedicated cable-connections to every location a PowerMeter is installed. Therefore, wireless solutions are the preferred choice for such environments. The ESP32 supports the WiFi standards 802.11 b/g/n with a bandwidth under ideal conditions of up to 130 Mbit/s according to the official documentation [152]. However, it was found that under real-world conditions (i.e., larger distance to the access point and multiple devices in the same network), only up to 12 Mbit/s could be achieved and maintained with TCP connections. This bandwidth is still far more than required to successfully send uncompressed voltage and current waveform data at the maximal sampling rate. This requires a minimum of $f_s \cdot \#measures \cdot \#bits\_per\_measure$ resulting in $7875 \cdot 2 \cdot 32\,\text{bit/s} = 504\,\text{kbit/s}$.

4.3.4 Firmware

The PowerMeter's firmware implements a lot of the functionality of the SmartMeter's firmware. Upon request, the PowerMeter reads the ADC values in a continuous loop and converts the raw 24-bit measurements to 32-bit float values representing the actual voltage and current measurements. Calibration parameters for voltage and current are stored in the non-volatile memory. Instead of ADC-triggered interrupts, a hardware timer of the microcontroller is configured to trigger an interrupt corresponding to the desired sampling rate (whenever the counter reaches $240\,\text{MHz}/f_s$ ticks). For each interrupt a new measurement is latched from the ADC via SPI. The sampling rate is again corrected using the RTC by adjusting the number of samples each second (cf. C5). A continuous data stream can be requested by a sink via TCP or UDP connection. Alike the SmartMeter, flow control mechanisms and packet numbers maintain data integrity and minimize data loss. The implemented watchdog mechanism resets the PowerMeters whenever a critical error occurs or the network connection drops. The firmware further allows to switch the connected appliance *on* or *off* upon an external request. This allows to integrate the PowerMeters into an existing smart home infrastructure. Further-

more, it allows to record voltage and current waveforms of appliance switch-on or switch-off events with precise control over exactly when these switching events happen. The WHITED [22] and PLAID [23] datasets contain isolated appliance switch-on events. The WHITED data was retrospectively shifted such that the switch-on event occurs at exactly 100 ms. Picon et al. [153] recorded the COOLL dataset which includes appliance switch-on events for different switch-on phase shifts, i.e., the time after a positive zero-crossing of the voltage signal. As PowerMeters have direct control over when the connected appliance is switched on, the recording of similar datasets would also be conceivable.

4.3.5 Modular Expandability

Similar to the SmartMeter, the PowerMeter includes an expansion header. This header is connected to the microcontroller's hardware serial interface allowing to add further functionality to the system without the need of developing a complete redesign of the hardware.

In a first attempt to exploit this interface, a sensor board was developed. Figure 4.9 shows the sensor PCB stacked on the PowerMeter PCB (left) and the complete system integrated into the housing (right).

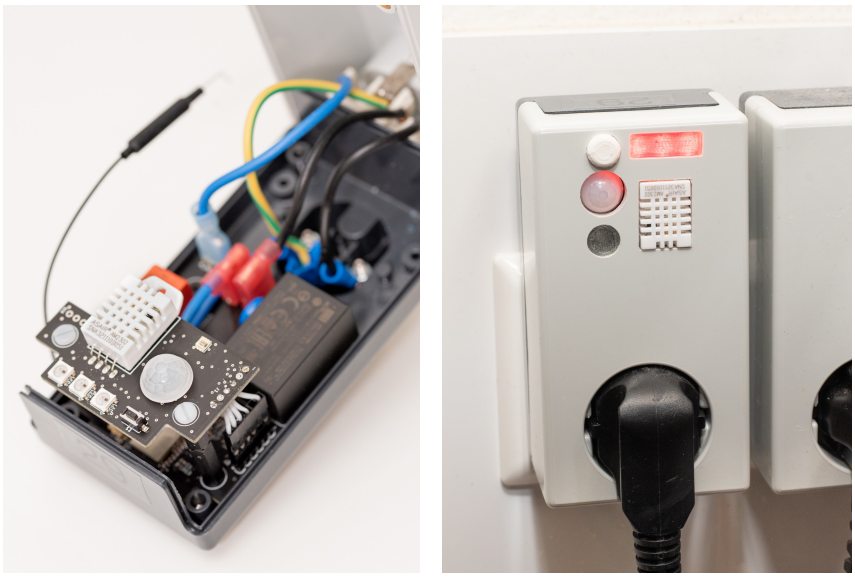

Figure 4.9: Sensor board stacked onto a PowerMeter (left) and the board integrated in the housing (right).

It incorporates typical environmental sensors including a temperature, a humidity, and a light sensor as well as a Passive Infrared Sensor (PIR) (to detect people's

movements in a room). The board further includes a physical button to directly switch the connected appliance *on* or *off* and LEDs for instant eco-feedback.

4.3.6 Evaluation

The PowerMeter is capable of measuring voltage and current waveforms with a maximum sampling frequency of 7.875 kHz. Even at this data rate, the corresponding measurements can still be sent to a sink via WiFi without continuous data loss (cf. C2, C3, and C7). According to the datasheet [149], the current value corresponding to the LSB is calculated as

$$LSB_I = \frac{V_{ref}}{cal_I \cdot A_I \cdot 2^{23} \cdot k_S}\,. \tag{4.2}$$

Given, the reference voltage $V_{ref} = 1.18\,\mathrm{V}$, a calibration parameter $cal_I = 0.875$, the resistance of the shunt $k_S = 3\,\mathrm{m\Omega}$, and the maximum current amplification of $A_I = 16$, the theoretical minimal detectable current is $3.35\,\mu\mathrm{A_{peak}}$ with a full-scale current of $28.1\,\mathrm{A_{peak}}$. Similarly, the minimum detectable voltage (corresponding to the value of the LSB) can be calculated alongside as

$$LSB_V = \frac{V_{ref} \cdot (1 + R_1/R_2)}{cal_V \cdot A_V \cdot 2^{23}}\,. \tag{4.3}$$

With the resistances of the voltage divider $R_1 = 810\,\mathrm{k\Omega}$, $R_2 = 470\,\Omega$, a calibration constant $cal_V = 0.875$, and an amplification factor of $A_V = 2$, the theoretical minimal detectable voltage is $138.6\,\mu\mathrm{V_{peak}}$ with a full-scale voltage of $1162.7\,\mathrm{V_{peak}}$.

Since the datasheet lacks information about the ADC's SNR, a test recording was conducted. A linear load with around 2000 W was connected and mains voltage with $230\,\mathrm{V_{RMS}}$ and a fundamental frequency of 50 Hz was applied. The corresponding voltage and current data measured by the PowerMeter was analyzed in the spectral domain using *Welch's* method [154]. This method uses overlapping windows and computes sub-spectra for each window. Finally, all sub-spectra are averaged, resulting in what is called a *periodogram*, which is a cleaner representation of the spectrum compared to using a single Fast Fourier Transform (FFT). In particular, multiple 8192 point FFTs are used with 10 % window overlap. To prevent spectral leakage and compress side lobes, an energy normalized hamming window was applied. The resulting periodogram of the current measurements is shown in Figure 4.10. The analysis results in a comparably good SNR of around 75 dB.

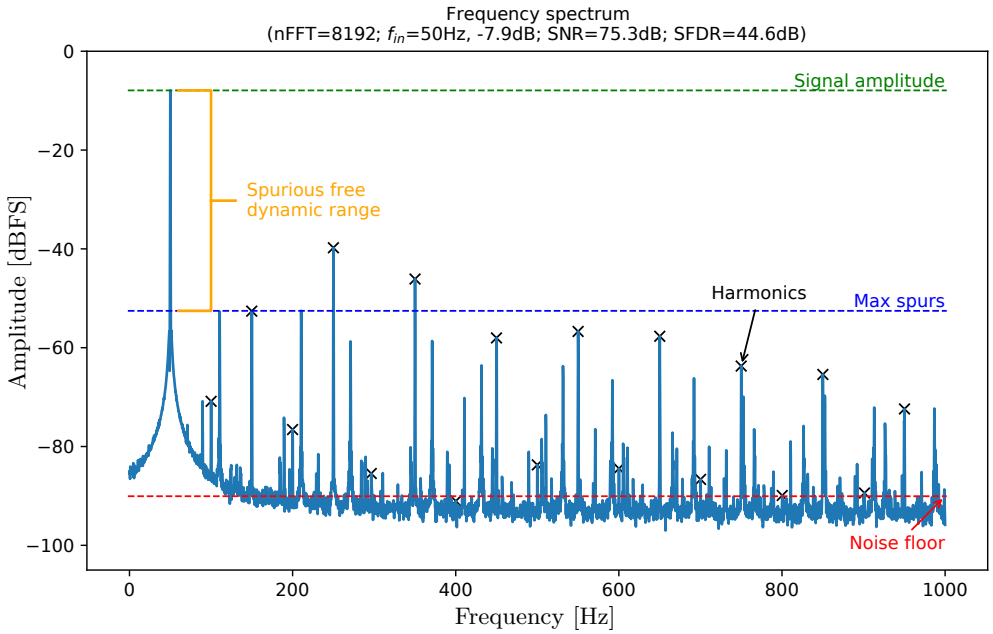

Figure 4.10: Periodogram of a test recording using a PowerMeter. The harmonics of the fundamental frequency ($f_0 = 50\,\text{Hz}$) are marked by ×. The maximum signal amplitude ($-7.9\,\text{dB}$) is highlighted by the dashed green line, the average of the noise floor ($-83.2\,\text{dB}$) by the dashed red line, and the spurious free dynamic range ($44.6\,\text{dB}$) is marked in yellow.

Similar test recordings as for the SmartMeter have been performed for the PowerMeters using different electrical loads. Figure 4.11 shows an excerpt of a test measurement including an espresso machine with around 1200 W. The measurements highlight that the PowerMeter is also capable of measuring comparably small electrical consumers (see switch-on event in Figure 4.11), without loosing the ability to measure typical high-power household appliances (see *heating* event in Figure 4.11). Therewith, the PowerMeters also meets challenges C3 and C2.

The power consumption of each PowerMeter itself is around 0.56 W. This is relatively low compared to the consumption of most appliances present in typical households. It is further comparable to the consumption of off-the-shelf smart plugs which have already been used to record NILM datasets. For instance, the *Plugwise* system, which has been used to collect the UK-DALE dataset [75], consumes around 0.5 W per plug.

Furthermore, the cost of a single PowerMeter is comparatively low with approximately €35 per unit, especially considering the flexibility it provides compared

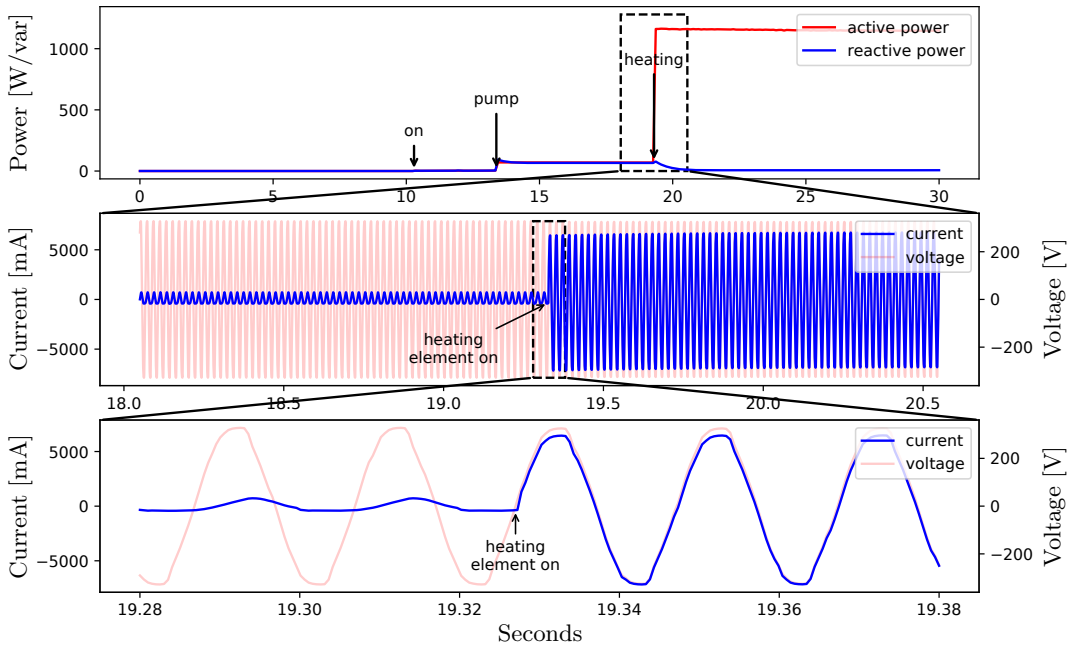

Figure 4.11: Test recording of an espresso machine during start-up using the PowerMeter hardware.

to off-the-shelf solutions. It is conceivable that the price could be reduced even further if the meters would be produced in higher volumes.

4.4 Additional Sensors

The framework depicted in Figure 4.1 allows to record arbitrary sensor values or other modalities simultaneously with the electricity measurements using an MQTT-API. MQTT [155] provides a standardized publish-subscribe messaging system and has emerged to one of the standard protocols in the world of IoT. If an instance wants to share information, it can send a message for a given *topic*. If other instances are interested in this information, they can *subscribe* to the specific topic. Each new message *published* under a certain topic is relayed by a broker to all instances which subscribed to this topic. MQTT builds on top of the TCP network protocol which guarantees the successful transmission of data. The recording PC (cf. Figure 4.1) hosts a central MQTT broker. A dedicated listener (software running on the recording PC, see Section 4.6) waits for incoming messages under a general topic *recording* and handles the conversion of incoming data into Comma-Separated Values (CSV) and the storage into *.csv* files. If a sensor module is

intended to be added to the recording, it simply needs to connect to the broker and send its data on a unique sub-topic (e.g., *recording/livingroom_temp*). Data must follow the JSON format. Each JSON key corresponds to a header entry in the resulting CSV file. A timestamp is added by the listener for each entry if the key *ts* is not present in the data. An example for a valid message of a temperature sensor is *recording/livingroom_temp {"value": 20.5}* representing a room temperature of 20.5 °C in the livingroom.

Three examples of additional modalities and sensors, which have already been successfully integrated into the framework, are representatively explained in more detail below.

Smart lighting:
Many light bulbs are nowadays substituted with smart light bulbs. Most of these can be controlled via a *ZigBee* gateway. Such a gateway can be incorporated to pass information when a light bulb changes its state, dimm setting, or light color. A *Python* script was implemented which interfaces with such a gateway to log the state changes of all light bulbs connected to the gateway using the MQTT-API. This allows to derive power consumption estimates without intrusively metering every light individually.

Sensors:
Figure 4.12 highlights how custom sensors can be embedded into the recording framework using the provided MQTT-API. In this flow diagram, an ESP32 is highlighted as an example data processor which can be directly connected to the MQTT broker as it has WiFi built-in. The ESP32 further provides certain inter-system interfaces such as *SPI*, *I2C*, or *UART*. This allows to rapidly prototype different sensors like temperature or occupancy sensors.

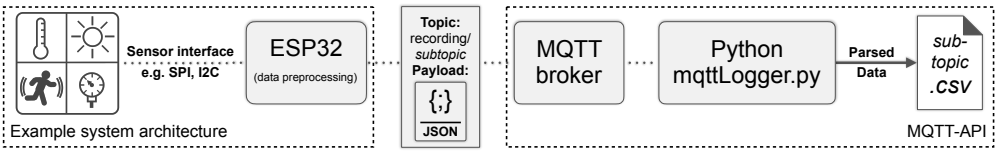

Figure 4.12: Example of extending the recording system by logging additional sensor data using the MQTT-API.

Bridges:
A similar structure as shown in the system overview of the sensor (see Figure 4.12) can be used to develop different gateways. As an example, a 433 MHz gateway was developed that logs state changes of switchable sockets, wall switches, or remote button-presses of devices that are equipped with 433 MHz. In addition, an infrared sniffer was developed which can receive commands from off-the-shelf

IR remotes. These commands can then be logged to MQTT and later encoded so that the logs indicate a corresponding button press of the remote. This allows to capture interactions with and state information of televisions, HiFi systems, or air conditioners.

4.5 Clock Synchronization

In Section 4.1, challenge C5 has been defined as "simultaneous recordings with high temporal accuracy". As the framework is comprised of numerous meters and sensors distributed across a home, a clock synchronization technique is required to maintain precise timestamps for the measured data and all information inferred from it. In initial experiments, non-negligible clock drifts of up to 300 ms could be observed after only 10 minutes. These are originating from clock inaccuracies of the used ADCs and microcontrollers. These varied depending on the individual meter device (SmartMeter and PowerMeter) and over time also for the same meter due to temperature and aging effects [156]. To reduce these drifts, each metering device was equipped with an RTC to synchronize the internal ADC clock. In addition, the Network Time Protocol (NTP) is utilized to synchronize the internal system time of a meter with an NTP server frequently. An NTP synchronization interval of 120 s was empirically found to be sufficient. As NTP also takes the Round-Trip Time (t_{RTT}) of the request itself into account, it can be accurate up to $\pm t_{RTT}/2$ [157]. If the NTP server is located in the same local network, it was found that t_{RTT} is constantly below 10 ms, resulting in a time synchronization accuracy of around 10 ms. Since t_{RTT} can be measured directly, only NTP requests that have response times better than $\pm$10 ms are used to synchronize the internal system time. If a time drift is detected after a successful NTP synchronization, it is slowly phased out by removing or adding samples using nearest neighbour interpolation. As only a maximum of a single sample is removed or added per second, only a minor jitter of $1/f_s$ seconds is added to the data (e.g., 125 µs for data which are sampled at 8 kHz).

4.6 Recording Manager

A Python script (*recordingManager.py*), which runs on a dedicated PC inside the home network, orchestrates the recording of the SmartMeter, each PowerMeter, and all additional sensor data. Since this PC does not require huge amount of processing power, a low-power single-board solution such as a Raspberry Pi [89] was found to be sufficient. The overall flow of the manager script is shown in

Figure 4.13. It consists of multiple sub-processes which run concurrently as separate threads on the recording PC. Each sub-process is further explained in the following.

Maintenance:
For reliability reasons, all meters (SmartMeter and all PowerMeters) are reset each day at midnight.

Memory watchdog:
The amount of data which are produced by the electricity meter is quite huge. For a setup with one SmartMeter and 20 PowerMeters sampling at a rate of 8 kHz and 2 kHz, respectively, approximately 35 GB of data are produced per day. Therefore, a watchdog checks the amount of available disk space each hour and notifies the user if disk space is critical.

File watchdog:
For backup purposes, each newly generated file is synchronized with an external data center. This is assumed to have *infinite* storage.

Statistics:
To stay updated on the recording status, the framework sends a small statistical report every 24 hours containing information about the home's energy consumption as well as information about the data sampling process (e.g., the number of meter dropouts, the successfully received data packets, and the estimated time drifts of each meter).

Connection manager:
To congregate the measurements, a central TCP server is hosted to which all electricity meters automatically connect upon start-up or restart. If a new meter connects to the server, the type and device ID of the meter is passed. If a new meter is identified, the script triggers an NTP synchronization with that meter. After a successful synchronization, the sampling process is initiated, i.e., a new thread of type *meter listener* is started. The NTP synchronization accuracy is crucial and forced to be better than 20 ms to continue. The start time has to be as precise as possible, since all future timestamps of the data are derived relative to the timestamp of the initial sample.

Meter listener:
The thread triggers the meter to start the sampling of measurements and creates the necessary file structure. If new meter data fails to be received for a certain amount of time or if a received chunk of data does not conform the correct format, an error is reported to the user. The received data are stored as Matroska multimedia container into sub-folders named after the corresponding meter (e.g., *smartmeter001*). The advantages of using multimedia containers to store curated time series data have been elaborated by Scholl and the author of this thesis

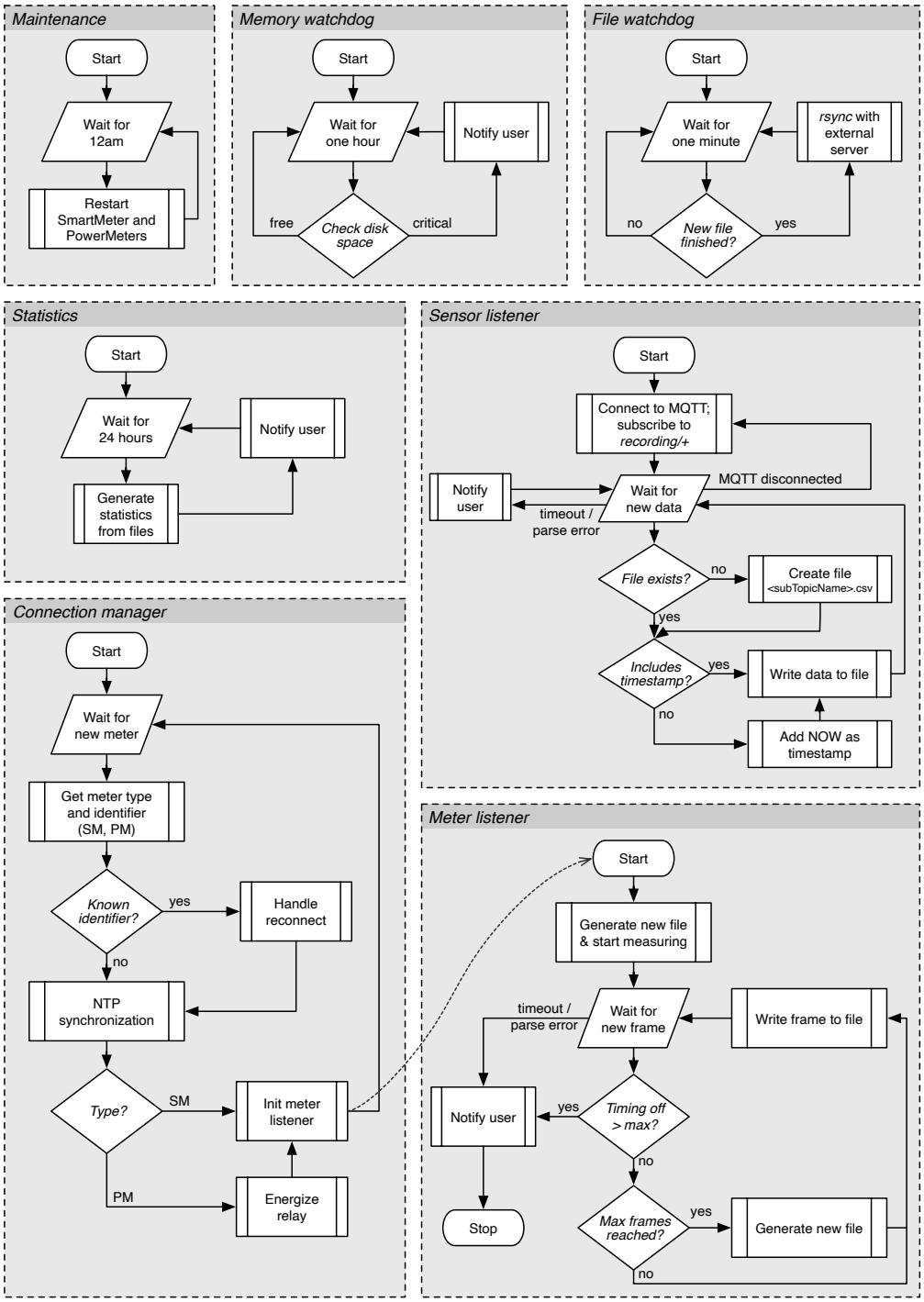

Figure 4.13: Individual tasks of the recording manager including several watchdogs for available memory, files not yet backed up, new statistics, or maintenance. A connection manager handles connecting SmartMeters (SM) or PowerMeters (PM), whose data are handled and stored by a separate listener thread for each meter. A sensor listener thread logs all MQTT sensor messages to file.

in [BC19]. While being optimized for audio or video streams, these containers allow to store regularly sampled sensor data as time-synchronized audio streams. Furthermore, multiple streams (video, audio, or subtitles) can be merged into a single file. Time series sensor data can, therewith, be stored as audio streams with text-based ground truth labels as subtitles. Additional metadata can also be stored for each stream inside the container.

The voltage and current measurements obtained by a meter are stored as a single *WavPack* [158] encoded audio stream inside a multimedia container. The stream has multiple channels for the voltage and current signals. SmartMeter data has six channels (v_{L1}, i_{L1}, v_{L2}, i_{L2}, v_{L3}, i_{L3}), while PowerMeter data has only two channels (v, i). As stated in [BC19], WavPack allows a lossless reconstruction of the data while maintaining high compression rates for time series data. In particular, a compression ratio of 1.46 was achieved for the voltage and current measurements using WavPack while a ratio of only 1.42 was achieved with *hdf5* [159] which has been used e.g., for UK-DALE [75] and BLOND [21].

Different metadata are also stored for each of the streams including the start timestamp (with microseconds resolution), the *id* of the particular meter, the sampling frequency, codec information, the name of the measured attributes, and the stream's duration. Thus, each file is self-descriptive and can be used without prior knowledge. File size and, therewith, file loading times are kept reasonable by splitting all files at regular time intervals. Furthermore, the local time of the first sample is appended to each filename. Amongst other parameters, the sampling rate, the measures taken, the storage location, the time interval at which the files are stored, and the file name format can be configured to the needs of the user prior to recording.

Sensor listener:
As already mentioned in Section 4.4, arbitrary sensor data or other modalities can can be stored with the framework if these data are published to a corresponding MQTT topic. The *sensor listener*, therefore, connects to the broker (also hosted at the recording PC) as a separate MQTT client. This client subscribes to all sub-topics under the topic *recording* and stores the incoming data into corresponding files named according to the respective sub-topic. If the data, which must be in JSON format, has no timestamp attached, the current system time of the recording PC is added to the data. If unexpected data are received or a sensor has not sent data for a certain amount of time, the user is notified.

User notification:
Kelly and Knottenbelt used a so called *babysitter* program to constantly monitor the status during the recording of the UK-DALE [75] dataset. Upon any error, their program sends an error report to a dedicated email address. Furthermore,

a lifeness message is sent each day. The message contains information about the home's power consumption during the last day. Likewise, the recording manager sends *user notification(s)* including errors or recording statistics to a configurable email address.

4.7 Post-Processing

In order to use the voltage and current measurements, no data pre-processing is required, as both the SmartMeter and all PowerMeters directly calculate the physical quantities (*Volt* for voltage and *Milliampere* for current measurements) from the raw ADC samples (cf. C7). In order to compress the information of the high-frequency voltage and current measurements, active (P), reactive (Q), and apparent (S) power are calculated in a post-processing step after the recording. The power is derived from the voltage (V) and current (I) waveforms based on the mains frequency f_l (e.g., 50 Hz in Europe) according to Equation 2.9, 2.10, and 2.11. Since data of commercial smart meters can typically be retrieved at a rate of 1 Hz to 0.01 Hz, an additional 1 Hz version of the power data is provided. The 50 Hz power is downsampled to 1 Hz by calculating the average of each block of 50 samples. The derived 50 Hz and 1 Hz power are stored for each meter individually and contain one day of data. Mains cycles for which the power could not be calculated, as no voltage and current data are available, are marked with a power of constant zero to maintain an equidistant time period between samples.

4.8 Discussion

The SmartMeter and PowerMeter hardware detailed in Section 4.2 and Section 4.3 allow to record electricity data of multiple appliances simultaneously (cf. C1) with a high sampling rate (cf. C2) over a long time period (cf. C3). With NTP synchronization and ADC sampling correction using RTCs, the data are of high temporal accuracy (cf. C5). Both systems are encapsulated in fireproof housings and all physical interfaces have been galvanically isolated from the mains voltage levels (cf. C6). The recording manager detailed in Section 4.6 orchestrates the dataset recording, stores all data into files, and notifies the user upon errors (cf. C7). An additional post-processing step calculates typical electrical quantities in more convenient time resolutions for quick data inspection (cf. C7). The MQTT-API allows to add other modalities such as sensor values or appliance state changes (cf. C4) using additional sensors or bridging interfaces either implemented as additional standalone sensors or as a software solution running on the recording PC (cf. C7).

Event-based NILM systems, however, require detailed information about each electrical event e.g., the light in the fridge switched on because the door was opened (cf. C4). While it is technically possible to obtain such information using dedicated sensors attached to every appliance (e.g., using a light sensor inside the refrigerator), this is highly impractical due to the versatility and sheer number of appliances that are being used. As an alternative to a dedicated monitoring of all states of each appliance, the electricity measurements can be manually inspected and labeled after the recording. However, the labeling of numerous hours of data by hand is tedious, time consuming, and prone to errors due to fatigue [99]. To tackle this challenge (C4), the framework incorporates a semi-automatic labeling technique as depicted in Figure 4.1 which is conducted after the recording of the dataset. This is explained in more detail in Chapter 5.

5 Data Labeling

This chapter addresses the challenge of dataset labeling. The employed techniques are enclosed in the contributions of [J21a], [C19a], and [W20]:

- Development of a semi-automatic labeling algorithm to be applied post-recording (see Section 5.1) [J21a, C19a].
- Design of a custom labeling tool for electricity data (see Section 5.2) [W20].
- Experimental evaluation of the semi-automatic labeling pipeline for two electricity datasets (see Section 5.3) [J21a, W20].

5.1 Automatic Event Extraction

Evaluating event detection algorithms or event-based NILM algorithms requires ground truth data for the recorded events contained in the dataset. The authors of the UK-DALE [75] dataset, therefore, recorded appliance turn on/off events for house 1 using switchable sockets. If a resident pressed the button on such a switchable socket, the current timestamp, appliance, and state of the socket (*on* or *off*) are logged. In particular, three disadvantages of such an approach are:

(1) Appliances that are hardwired to the mains like the stove or lighting cannot be equipped with such a socket.

(2) Only on/off events can be logged. Most household appliances are multi-state appliances that have more than just a binary state *on* or *off*.

(3) Appliances that change their state without user interactions cannot be labeled (e.g., a kettle turns off automatically if the water is boiling).

The authors of the BLUED [94] dataset added additional sensors to appliances (e.g., light sensors to the lighting) to log appliance events. The drawbacks of this approach are that obtrusive and potentially battery powered sensors are added and have to be maintained. Furthermore, these sensors have to send data wirelessly to a sink, requiring additional clock synchronization so that the appliance event timestamps can be mapped to the electricity data.

Instead of relying on additional sensors, the approach detailed in this chapter is based on the subsequent labelling the data after the recording. A similar approach has already been successfully applied to the domain of electricity data by Pereira [39]. Since Pereira's method does not support textual labels and generates a comparably large number of false positives if low detection thresholds are used, several strategies were incorporated to improve his algorithm. Overall, the proposed labeling algorithm consists of three steps: (1) *event detection*, (2) *high variance filtering*, and (3) *unique event identification*.

5.1.1 Event Detection

Event detection describes the process of identifying relevant changes in the electricity data which correspond to certain appliance state changes (cf. definition of the term event in Section 3.3.1). A probabilistic event detector based on the LLR test introduced by Pereira [39] is used. The detector is further enhanced by an adaptive threshold technique. This detector first calculates the likelihood ($L[i]$) that an event has occurred at sample i using a detection window over the apparent power signal ($S[i]$). The detection window splits into two sub-windows, the *pre-event* window $[i-a, i[$ and the *post-event* window $[i, i+b]$. The window widths are determined by the parameters a and b, respectively. $L[i]$ is then calculated as

$$L[i] = \ln\left(\frac{\sigma_{[i-a,i[}}{\sigma_{[i,i+b]}}\right) + \frac{\left(S[i] - \mu_{[i-a,i[}\right)^2}{2 \cdot \sigma^2_{[i-a,i[}} - \frac{\left(S[i] - \mu_{[i,i+b]}\right)^2}{2 \cdot \sigma^2_{[i,i+b]}}\,, \tag{5.1}$$

where $\sigma_{[a,i[}$, $\sigma_{[i,b]}$, $\mu_{[a,i[}$, and $\mu_{[i,b]}$ are the standard deviations and means of the pre-event and post-event windows, respectively. Figure 5.1 shows the apparent power of a desktop fan recorded using a PowerMeter and the corresponding calculated likelihood $L[i]$ using Equation 5.1. An exemplary sliding window and the corresponding means (green) and variances (blue) are highlighted.

This signal L is further cleaned using an adaptive threshold ($thres_i$). If the change of the mean value from pre- to post-event drops below this threshold, $L[i]$ is forced to zero using

$$L[i] = \begin{cases} L[i], & \textit{if } \left|\mu_{[i-a,i[} - \mu_{[i,i+b]}\right| > thres_i \\ 0, & otherwise \end{cases}. \tag{5.2}$$

The adaptive threshold $thres_i$ is defined as

$$thres_i = thres_{min} + m \cdot \mu_{[i-a,i[}\,, \tag{5.3}$$

with the minimum power change of interest $thres_{min}$ and a linear coefficient m.

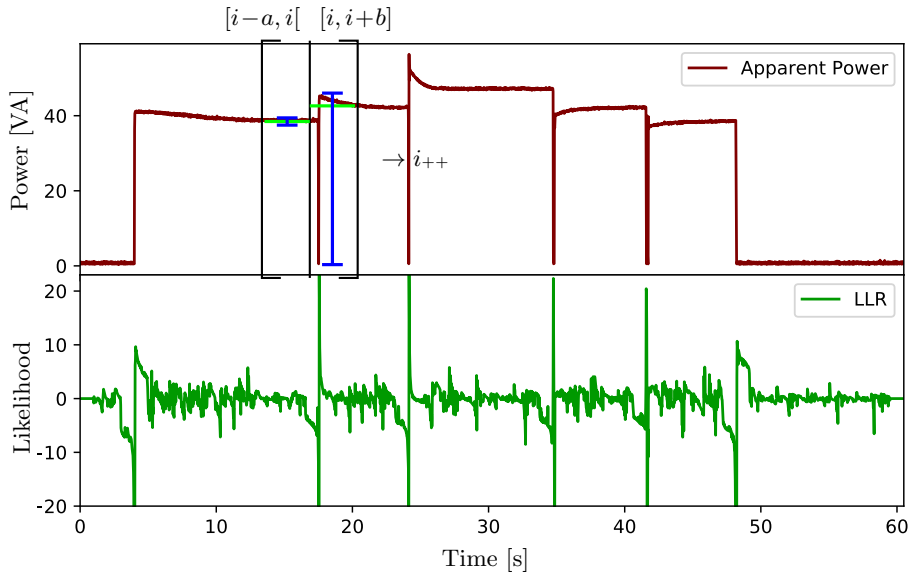

Figure 5.1: PowerMeter recording of a desktop fan. The LLR (see Equation 5.1) is shown in the bottom plot. An exemplary sliding window with the corresponding means and variances is shown in the top plot.

This coefficient causes a linear increase of $thres_i$ with the average power of the pre-event window. Typically, the variance in the power measurements is proportional to the magnitude of the power. This effect is caused by increasing noise in the appliance or the analog frontend of the electricity meter. If a fixed small threshold is set (e.g., $thres_i = 3\,\mathrm{W}$), a large number of false events may occur at regions where more power is drawn. If a fixed high threshold is set, low power events may be missed. Pereira and Nunes used a comparatively large threshold of 30 W in [99] to reduce the amount of false events. However, such a threshold does not allow to detect state changes of low-power appliances such as battery chargers or LED lighting. The linearly increasing threshold according to Equation 5.3 adapts to larger fluctuations and, therewith, helps to reduce the number of false events significantly. By allowing to set smaller minimal threshold values, the number of missed events can also be reduced. The resulting $L[i]$ after applying the adaptive cleaning procedure (Section 5.1.1) is shown for the same desktop fan recording in the lower plot of Figure 5.2 (see below for the parameters used).

If an event is detected at sample i, the likelihood will also be non-zero around that sample due to the sheer size of the pre-event and post-event windows, respectively. This can also be seen in Figure 5.1 and Figure 5.2. To identify the exact sample at which the event occurred, a *voting window* is layed around adjacent samples for which $L[i]$ is non-zero. For each of these voting windows, the sample correspond-

ing to the maximum absolute value of $L[i]$ is identified. The minimum distance between two events is further restricted by an additional parameter (l). If an event has been identified at a sample corresponding to time t_e, all other events within $[t_e, t_e + l]$ are removed.

Therewith, the proposed algorithm has five adjustable parameters. The duration of the pre-event and post-event window, the minimum detection threshold $thres_{min}$, the linear coefficient m, and the minimum distance between events l. A user should specifically adjust the parameters $thres_{min}$ and l according to prior knowledge of the data: a low threshold $thres_{min}$ is required if events with small mean changes are expected, and a short l should be chosen if events can happen close in time. Values that seem to work quite well across different appliances are: pre-event window $= 1\,\mathrm{s}$, post-event window $= 1.5\,\mathrm{s}$, $thres_{min} = 3\,\mathrm{W}$, $m = 0.005$, and $l = 1\,\mathrm{s}$.

5.1.2 High Variance Filtering

Appliances such as PCs or televisions draw variable power depending on the current context (e.g., the current computational load of the PC or the content on the TV screen [160]). This can causes a large number of false events using the LLR test. To filter out these false events, signal regions are identified that have a high variance. All events found in such regions are removed. This is achieved by calculating the mean ($\mu[i]$) and variance ($\sigma[i]$) over a sliding window. If $\sigma[i]$ is larger than $n \cdot \mu[i] + thres_{min}$, the window is marked. If the length of consecutively marked windows exceeds a certain length (w), all events in these windows are removed. Specific values for w and n which tend to remove false events while still keeping relevant events across different appliance types were found empirically as $w = 4\,\mathrm{s}$ and $n = 0.01$.

5.1.3 Unique Event Identification

Similar events of an appliance are identified using unsupervised clustering. The fact is utilized that most appliances draw different but constant power before and after an event (e.g., the kettle before and after it is switched on). The different power levels represent internal states of the appliance (e.g., *off* and *on* for the kettle). Depending on the internal electrical components, appliances can easily have way more than two unique states (see e.g., the derived states of the desktop fan shown in Figure 5.3). To identify different power levels, the data are split at each event and the mean power demand between these splits is calculated. Unique power levels (representing unique appliance states) are then identified using hierarchical clustering with a distance threshold determined by $thres_{min}$. Each

cluster is given a textual *ID* which is used to assign a label to each event (*S0*, *S1*, ... as shown in Figure 5.2). Since some appliances show a higher rush-in power followed by a power settling (see e.g., the power during the transition from *S2* to *S3* in Figure 5.2), 10 % of the highest and lowest power samples are removed between each event before a corresponding mean value is calculated.

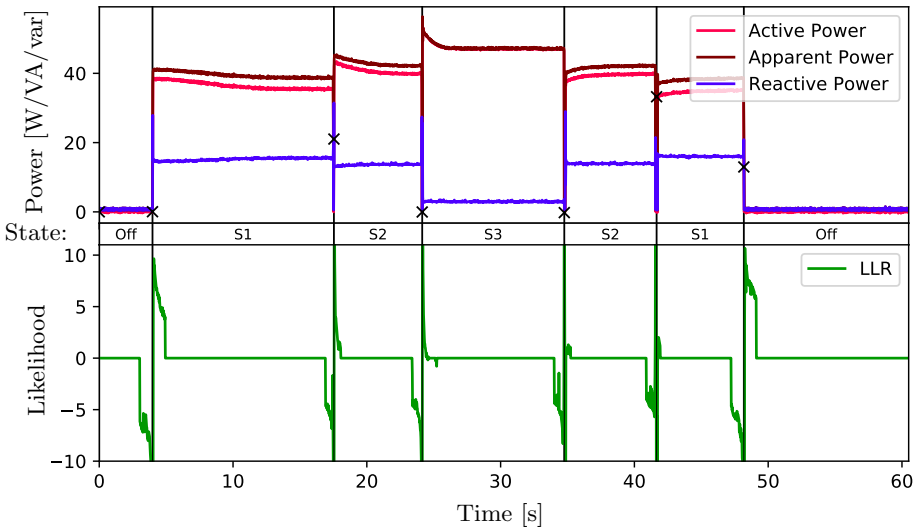

Figure 5.2: Recording of a desktop fan (top); calculated Log-Likelihood Ratio (bottom); recognized events are shown using vertical black lines and the clustered states are highlighted between consecutive events.

By using the event extraction algorithm, an appliance power signal can be pre-labeled. Each detected event that remains after filtering is marked and a label corresponding to the mapped event cluster is assigned. The overall process is shown in Figure 5.2 for power measurements of a desktop fan. Besides adding labels, the algorithm can also be used to automatically generate Finite State Machine (FSM) models of the recorded appliance. Figure 5.3 shows the automatically generated FSM model of the desktop fan. These models are constructed by storing the unique clustered events as states and analyzing the event sequence for state changes, which are then regarded as edges between these states.

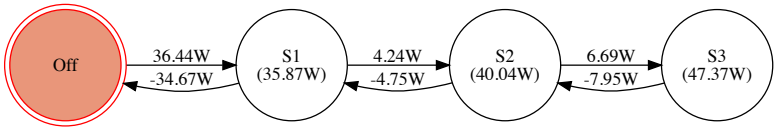

Figure 5.3: Appliance model automatically generated from the apparent power signal shown in the top plot of Figure 5.2.

5.2 Annoticity Tool

To combine the automatic event labeling presented in Section 5.1 with a graphical human supervision, the Annoticity inspection and labeling tool was created. The tool was specially designed for the task of generating precise text-based ground truth labels for events in electricity data but can also be adapted to annotate other time series data of any modality. As will become apparent, the tool can significantly reduce the labeling effort if paired with the automatic labeling algorithm. Annoticity is realized as a web application and provides direct access to various publicly available electricity datasets without the need to first download the dataset onto disk. Users can add labels manually, review automatically generated labels, or modify existing label sets. The labels can be downloaded in various file formats including an option to store the labels in the same file as the data. Manual labeling and inspection is performed on the client side while data fetching and automatic labeling is performed on the server side. The overall workflow is depicted in Figure 5.4.

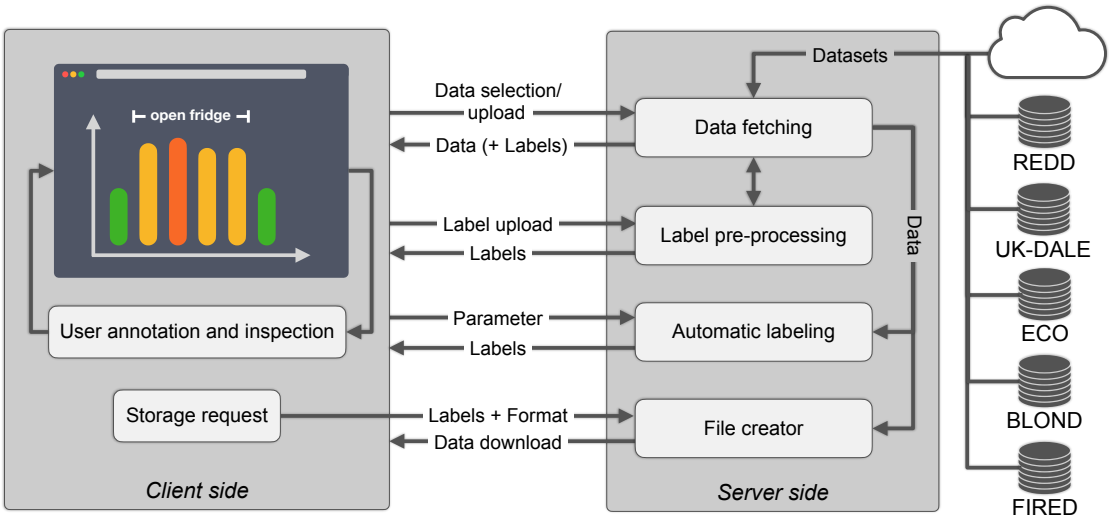

Figure 5.4: Flow of the Annoticity labeling tool. Data fetching, automatic labeling and file creation is performed on the server side, while manual labeling and user interaction is handled on the client side.

Annoticity has been made available to the public for everyone to discover electricity data or label datasets[1].

[1] `https://earth.informatik.uni-freiburg.de/annoticity`

5.2.1 Backend

The server backend is written in Python using the *Django* framework [161]. The backend's main purpose is to load the data and prepare it for visualisation, perform the task of automatic labeling, and provide file downloads. Data can be uploaded through the web application. Currently, Matroska [162] multimedia containers (*.mkv*) and *.csv* files are supported. Some publicly available datasets are directly accessible via selection: REDD [17], UK-DALE [75], BLOND [21], ECO [47], and FIRED [N20]. Depending on the dataset, either a copy of the data is stored on the server or the data are acquired via direct cloud access from the original source of the dataset. The backend resamples the data to a reasonable sampling rate according to the current time-span selected by the user. If the dataset already contains labels, those are by default displayed to the user. Additionally, a file containing labels can be uploaded and modified. The supported formats are *.csv*, *.srt*, and *.ass*.

5.2.2 Frontend

The client side is implemented in *HTML* and *JavaScript* and provides the frontend to the user. Annoticity's graphical user interface is shown in Figure 5.5. After either uploading a file or selecting a time-span and meter of an available dataset, the user can visually inspect the data. Different measures (e.g. active and reactive power) can be selected and data can be zoomed-in which leads to a data download at a higher sampling rate. The user can manually add a label by clicking at the signal's slope to mark an event, remove the label by double-clicking on the event marker, or modify the label by selecting its text. Each label consists of a timestamp and a (possibly empty) textual description. Labels are stored either as plain *.csv*, *.ass*, or *.srt* files, or embedded into a *.mkv* file together with the original data.

As Annoticity is designed as a web application, access to the data is provided from anywhere. The only requirements are a modern browser and a connection to the Internet. With the user management directly built into the Django framework, the Annoticity tool is already primed for collaborative labeling and gamification mechanisms in a future version (e.g., as realized by Cao et al. [137]).

5.2.3 Automatic Labeling

The automatic labeling, described in Section 5.1, can be applied to obtain event positions inside the selected data. The required calculations are performed on the backend as depicted in Figure 5.4.

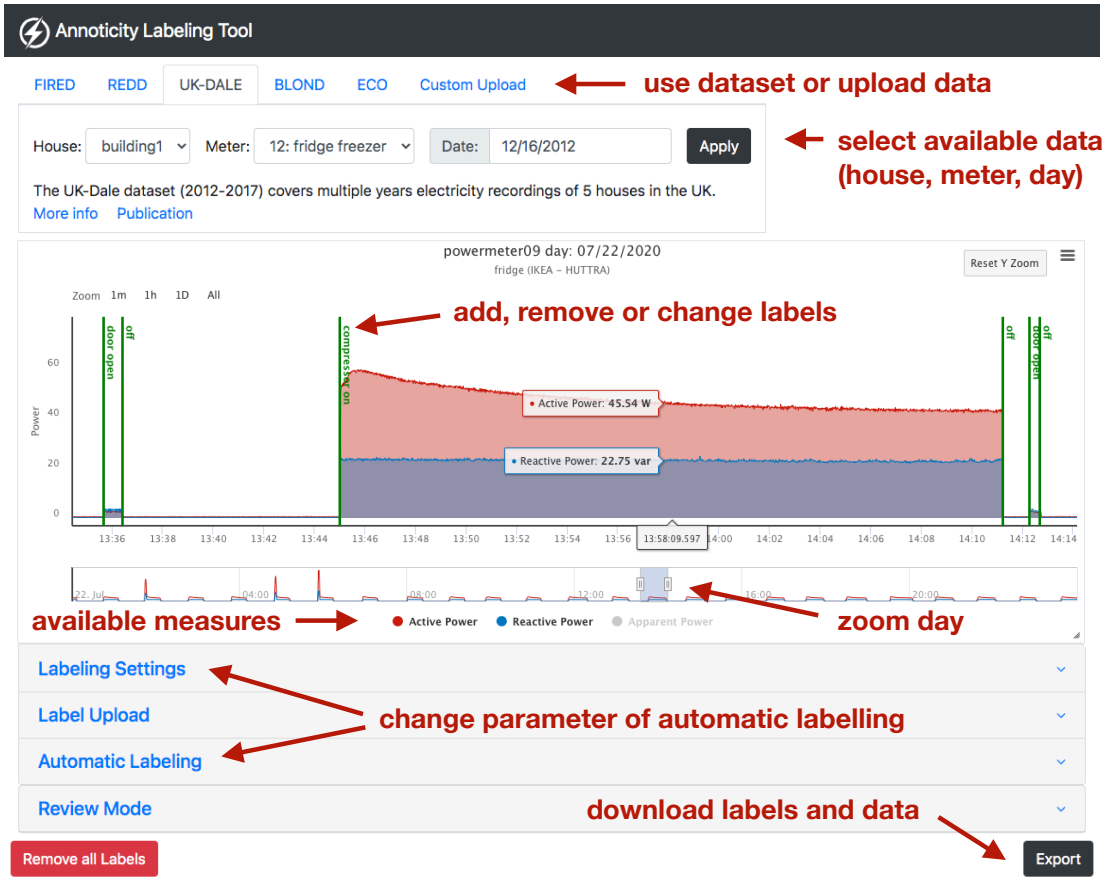

Figure 5.5: The graphical user interface of the Annoticity labeling tool. The user can select data and times of different electricity datasets at the top. Labels can then be added and adjusted either manually by interacting with the displayed electricity data, or by invoking the automatic labeling algorithm whose parameters can be adjusted at the bottom.

The frontend allows to set the parameters of the automatic labeling algorithm, namely the *pre-event* and *post-event window length*, $thres_{min}$, m, l, w, and n. Furthermore, the sampling rate to which the data are resampled before the algorithm is applied can be adjusted. As explained in Section 5.1, the algorithm is designed to be applied to the apparent power signal. It shows, however, similar results if it is applied to the active power signal (P). If neither apparent nor active power are available in the uploaded or selected data, but raw voltage and current waveforms are, apparent power (S) is calculated on demand. After the events have been identified, they are clustered, pre-labeled as shown in Section 5.1.3, and sent to the client side for inspection and validation. A special inspection mode further allows to quickly iterate through the labels at a proper zoom-level of 10 s.

5.3 Evaluation and Results

The proposed event detection algorithm (see Section 5.1.1) has been evaluated using the publicly available datasets REDD [17] and FIRED [N20]. In total, 14 days of FIRED and 8 days of REDD have been used, and 6323 labels have been added to the data. Annoticity was used to automatically generate an initial set of labels. To compare this set to an actual ground truth, the initial set was revised by an expert rater who visually inspected the data. False events were removed, missing events were added, and a distinct and descriptive label was assigned to each appliance state. Figure 5.6 shows both, the initial set of labels and the final labeled data of the *espresso machine* from the FIRED data. In particular, 4379 events for FIRED and 1944 events for REDD were generated by the expert rater to serve as the ground truth for the following evaluation. It is noted that the focus of this evaluation lies on event detection, hence, high variance filtering has not been applied and the unique state labeling was not examined in this evaluation.

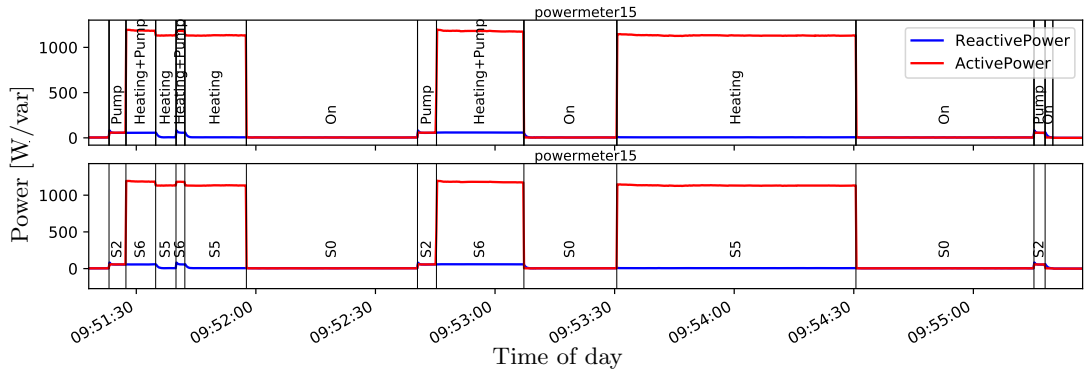

Figure 5.6: The fully labeled data of the *espresso machine*. The bottom plot shows the initial labeling of the automatic labeling algorithm, while the top plot shows the final labeling after human supervision. (The rightmost event has been missed by the algorithm.)

The results are presented in terms of the confusion matrix as values for TP, FP, and FN (see Section 2.5). A TP is defined as a detected event that is reflected within two seconds in the set of ground truth events. Accordingly, a FP is defined as a detected event without a corresponding event in the ground truth, and a FN is an event in the ground truth data which has not been found by the algorithm. The corresponding F_1-score was calculated to summarize these numbers into a single metric according to Equation 2.22.

5.3.1 REDD dataset

For REDD [17], apparent power measurements are available at mains, socket, and sub-circuit level at a sampling frequency of approximately 1/3 Hz. The evaluation uses eight days of socket- and sub-circuit-level data (from April 19 to 26, 2011) from house 2. The data has been resampled at a regular base of 1 Hz. Due to the low sampling rate of the REDD dataset, the following parameters were used for the evaluation: pre-event window and post-event window = 3 s, $thres_{min} = 4.5$ W, $m = 0.009$, and $l = 2$ s.

The results are shown in Table 5.1. Overall, an F_1-score of 86.73 % was achieved. 1627 out of 1944 events (83.69 %) were identified correctly. The *refrigerator*, which exhibits the most events, shows the highest number of FP. These stem from short defrosting cycles which have not been treated as relevant events during the manual ground truth labeling process. The *lighting* and the *kitchen outlets #2* show high numbers of FNs due to events close or below the used minimum mean change threshold of 4.5 W.

Table 5.1: Event detector performance on REDD. The *washer dryer* has never been used during the evaluation time period.

Appliance	Events	TP	FP	FN	F_1
Dishwasher	59	51	34	8	70.83
Disposal	19	11	0	8	73.33
Kitchen Outlets #1	72	49	3	23	79.03
Lighting	156	85	7	71	68.55
Stove	47	40	6	7	86.02
Microwave	158	130	20	28	84.42
Washer Dryer	0	0	2	0	0.00
Kitchen Outlets #2	525	464	5	61	93.36
Refrigerator	908	797	104	111	88.11
sum	1944	1627	181	317	86.73

5.3.2 FIRED dataset

For FIRED [N20], power measurements are available for 21 meters at a sampling rate of 50 Hz. Data of 15 appliances were selected and the event detection algorithm was evaluated on 14 days of data (from July 22 to August 4, 2020). Six appliances for which no distinct events were labeled manually (network equipment etc.) were omitted. The following parameters are used for the evaluation:

pre-event window length = 1 s, post-event window length = 1.5 s, $thres_{min}$ = 3 W, m = 0.005 and l = 2 s.

It was observed that the labeling algorithm performs quite well ($F_1 > 92\,\%$) for appliances which show distinct states in the power signal (such as the *oven*, *kettle*, or the *espresso machine* shown in Figure 5.6). For devices which draw variable power in between states (such as the two PCs or the *coffee grinder*) a large number of false events was triggered. It can be assumed that using a higher $thres_{min}$ or linear factor m as well as additional cleaning steps such as the high variance filtering step, explained in Section 5.1.2, would have reduced the number of false events significantly. To put the results into a different perspective, Table 5.2 shows the evaluation split into two groups representing appliances that show distinct states and appliances which draw variable power.

Table 5.2: Results of the event detection algorithm applied to the FIRED data; the results are evaluated for two appliance groups. In #1 appliances are grouped which have distinct steady states. #2 groups appliances that draw variable power. *Events* marks the number of ground truth events labeled manually.

Group	Appliance	Events	TP	FP	FN	$F1$
#1	Baby Heat Lamp	6	6	0	0	100.00
	Fridge	1006	863	2	143	92.25
	Coffee Grinder	348	250	114	98	70.22
	Espresso Machine	1880	1760	0	120	96.70
	Kettle	30	30	0	0	100.00
	Hairdryer	18	17	0	1	97.14
	Hifi System, Subwoofer	45	44	37	1	69.84
	Television	79	65	4	14	87.84
	Kitchen Spot Light	12	12	0	0	100.00
	Oven	138	138	1	0	99.64
	Fume Extractor	47	47	1	0	98.95
	Sum	3609	3232	159	377	92.34
#2	Smartphone Charger #1	96	83	1491	13	9.94
	Smartphone Charger #2	63	45	7999	18	1.11
	Office Pc	583	410	85367	173	0.95
	Media Pc	28	10	26632	18	0.07
	Sum	770	548	121489	222	0.89

The overall F_1-score of group #1 is 92.34 %. 3232 out of 3609 events are recog-

nized correctly. The *coffee grinder* and the *HiFi system* show a comparatively low performance with a high number of FP. This is due to higher variance when the grinder's motor is active or music is playing, and could have been avoided by using a higher linear factor m or a higher threshold $thres_{min}$. The *espresso machine* has very short heating cycles and a pump which together can cause events close in time. These could not be detected due to the chosen parameter $l = 2\,\mathrm{s}$ for the filtering step and account to the comparably large amount of FN. Still, the algorithm shows quite promising overall results for nearly all appliances of group #1. The results for appliances of group #2, however, show a very large number of false positives. These stem from shorter periods in which more power is consumed by the appliance. For instance, depending on the current calculations of a PC, more power can be required for a short period of time. As smartphones and PCs have comparable characteristics, equivalent (but smaller) power peaks are present in the data of the smartphone charger. This is especially apparent when the battery of the connected smartphone has been fully charged, since the power management circuit inside the smartphone then draws power directly from the charger instead of the internal battery. Nevertheless, way better results were obtained by including high variance filtering and by individually adjusting the algorithm's parameters for these appliances. While the latter requires additional manual effort, it is still faster and more convenient compared to a fully manual labeling. The better results for the FIRED dataset compared to REDD (if considering appliances of group #1) indicate that higher data resolution and sampling rate are beneficial for event detection methods.

5.3.3 Comparing the Labeling Effort

To get an overall estimate of how much labeling effort can be reduced by using the automatic labeling, the raw number of clicks required to mark all events from scratch was compared to the number of clicks required to supervise and modify the pre-labeled event set generated by the event detection algorithm. In total 6323 events were labeled. If the task of applying textual labels is omitted, marking the events would still have required at least 6323 clicks. If the appliances in group #2 of FIRED are skipped, the event detection algorithm automatically placed 4859 events at the correct position for REDD and FIRED. With 340 falsely classified events, 694 missing events, and the 770 missing labels of group #2 in FIRED (which would require manual labeling), 1804 clicks would have been required to remove false events and add missing events. Therewith, the sheer amount of clicks could already be reduced by 71.47 % not accounting for the support which is provided by the Annoticity tool if additional textual labels should be added to the events. If the fact that removing a misplaced label (t_{del}) usually takes less

time than manually adding a label from scratch (t_{add}) is taken into account, Equation 5.4 can be applied.

$$reduction = 1 - \frac{t_{add} \cdot \#(missed\ events) + t_{del} \cdot \#(false\ events)}{t_{Add} \cdot \#(all\ events)} \tag{5.4}$$

Using $t_{add} = 10\,\mathrm{s}$ and $t_{del} = 5\,\mathrm{s}$ as a reasonable guess for the corresponding times and time difference, the reduction in labeling effort is actually 74.16 % compared to a fully manual approach.

To further model the help provided by the Annoticity GUI, it is possible to look at appropriate metrics from user interface research. *Fitts's law* predicts that the time required to move to a target area is a function of the ratio of the distance to the target (D) to the target's width (W). An illustrative example is the selection of a checkbox with a mouse. The distance to the checkbox is D, while the width of the checkbox is W. Fitts, therefore, proposed the *index of difficulty* in [163] as

$$ID = \log_2\left(\frac{2 \cdot D}{W}\right) . \tag{5.5}$$

This model has further been embedded into the Keystroke-Level Model (KLM) introduced by Card et al. [164]. KLM consists of the six operators: (K) the time to press a keystroke or mouse button, (P) the time to point to a target with the mouse (cf. *Fitts's law* [163]), (H) the time to home the hand on the mouse or the hands on the keyboard, (D) the time to draw a straight line with the mouse, (M) the time for decision making, and (R) the system's response time. Several extensions have been proposed including the separation of keyboard presses and mouse clicks using the operator B for the latter [165]. Reasonable times for some of the different operators are proposed in [164] and [165] as: $K = 0.12$ for a good typist, $P = 1.1$, $H = 0.4$, $M = 1.35$, and $B = 0.1$.

According to KLM, manual event-labeling using Annoticity can be encoded as:

1. find an event in the measurements (M)
2. point to the event (P)
3. press and release the mouse button ($2B$)
4. move hand to keyboard (H)
5. type in the label text of the event ($n \cdot K$ with varying n depending on the specific text; e.g., $n = 11$ for "Turned On" including the shift key). This, however, might be reduced to $n = 1$, with one specific character for a unique event. The character can be later substituted by the corresponding label text in a single operation for all events with the same label text.
6. press enter (K)
7. move hand back to mouse (H)

It is noted that this simplified encoding does not account that it may be required to zoom into the data (and afterwards revert the zooming), which has been identified to take between 4 to 8 s. This will be denoted using $Z = 6.0$. According to this encodings, a complete manual labeling of all 6323 events using Annoticity would require around 17 hours:

$$\begin{aligned} T_{manual} &= \#events \cdot (M + P + 2B + H + n \cdot K + K + H + Z) \\ &= 6323 \cdot (1.35 + 1.1 + 2 \cdot 0.1 + 0.4 + n \cdot 0.12 + 0.12 + 0.4 + 6.0) \\ &= 60511.11 + 758.76 \cdot n \\ &= 61269.87 seconds \quad \text{for } n = 1 \end{aligned} \tag{5.6}$$

In contrast, the removal of an event using Annoticity's *review mode* is encoded as:

1. decide if the event is a false event (M)
2. point to the false event (P)
3. press and release the mouse button twice ($4B$)
4. point to the *next event button* (P)
5. press and release the mouse button ($2B$)

If the event is a true event, steps 2, 3, and 4 are not required. In the review mode, no additional time for zooming is required, as the measurements are already presented at a proper zoom level. The time required by Annoticity to automatically label the data depends on the selected resolution and the specific data. But it is usually less than $R_1 = 8.0$ s for one day of data sampled at 50 Hz and less than $R_2 = 0.5$ s for 1 Hz. According to this encoding, the semi-automatic labeling of the events, including the review of 4859 true events, the deletion of 340 false events, and the manual addition of 1464 missing events, would require around 7 hours:

$$\begin{aligned} T_{semi} &= 11 \cdot 14 \cdot R_1 + 9 \cdot 8 \cdot R_2 + 4859 \cdot (M + 2B) \\ &\quad + 340 \cdot (M + P + 4B + P + 2B) \\ &\quad + 1464 \cdot (M + P + 2B + H + n \cdot K + K + H + Z) \\ &= 25480.61 seconds \quad \text{for } n = 1 \end{aligned} \tag{5.7}$$

This results in a time reduction of 58.41 % when using Annoticity with semi-automatic labeling compared to using it without.

Considering that no filtering was applied for the labeling algorithm and a fixed parameter set was used per dataset to simplify evaluation, it can be assumed that the number of FPs and FNs can further be reduced if filtering is applied and the parameters are adjusted individually for a particular appliance. Moreover, as the labeling tool also identifies identical events, the additional workload of adding textual labels is reduced to specifying the name of the corresponding event clusters (e.g., *compressor on*, *door open*, *off* for the fridge as shown in Figure 5.5).

6 FIRED Dataset

This chapter presents the Fully-labeled hIgh-fRequency Electricity Disaggregation (FIRED) dataset. FIRED is the first residential electricity dataset which features high-frequency aggregated- and appliance-level recordings with additional ground truth event labels. The dataset was highlighted in the contributions of [J21a] and [N20]. This chapter includes:

- An in-depth explanation of the data included in the dataset (see Section 6.2).
- A statistical analysis of the recorded data (see Section 6.3).
- A technical validation of the data in terms of time synchronization, data coverage, and validity of the measurements (see Section 6.4).
- Information about how to access and how to use the data (see Section 6.5).

Existing electricity datasets have been introduced in Section 3.2. Their shortcomings have already been discussed in Section 4.1 and need to be avoided for novel electricity datasets: (1) large gaps in the data, (2) low sampling rates for individual appliance data, (3) missing event information, (4) not all electrical consumers are known during recording, and (5) a complicated data loading process. In order to tackle these shortcomings, a set of design goals (or challenges) has been defined in Section 4.1 for the recording system of a dataset. These design goals go hand in hand with the desired characteristics of a versatile electricity dataset. Such a versatile dataset should include: Simultaneous aggregated and individual appliance data (C1) of high sampling rates (C2) over a long time period without larger gaps (C3). The data should include event labels (C4) with a high temporal accuracy (C5) and should be usable out-of-the-box (C7).

To meet these design goals, the framework introduced in Chapter 4 was utilized to record the Fully-labeled hIgh-fRequency Electricity Disaggregation (FIRED) dataset. FIRED includes 101 days days of aggregated three-phase current and voltage measurements sampled at 8 kHz as well as 21 time-synchronized individual appliance measurements sampled at 2 kHz from a residential apartment in Germany. Furthermore, it includes sensor readings such as room temperatures and additional state information of certain appliances and each light bulb in the apartment. The Annoticity labeling tool, introduced in Section 5.2, was used to fully label all state changes of the individually-metered appliances for a time period of two weeks.

6.1 Recording Setup

FIRED was acquired in a single three-room apartment with 79 m^2 of space (open combined kitchen and living room, bedroom, child's room partly used as office, hallway, bathroom, and storage room). The apartment is inhabited by two adults and one infant, and it is located in an apartment building consisting of seven apartments on four floors. The building was constructed in 2017, is heated via a district heating, and most rooms are equipped with air filters with built-in recuperators. According to the building's energy certificate, it requires a primary energy consumption of 12 kW h/m^2 per year. The apartment's power grid is a three-phase 50 Hz system consisting of L_1, L_2, L_3, and neutral (N) wires. L_{1-3} have a phase shift of 120°. Access to the apartment's electrical system is given through a fuse box located in the hallway. All lights installed in the apartment are off-the-shelf smart light bulbs with a built-in *ZigBee* module. This allows that the lights are turned on or off via a smartphone application, voice assistant, or regular wall-light-switch. It further allowed to log all state changes during the recording of the dataset using a ZigBee bridge as explained in Section 4.4. The washing machine, dryer, and freezer are located in the basement of the building, and are not part of the recording.

A SmartMeter (see Section 4.2) was installed in the apartment's fuse box. Split-core current transformers were attached to the three incoming supply legs. For voltage measurements L_1, L_2, L_3, and N were connected in parallel. The meter is supplied with power by an additional L_1 leg which is secured by a separate 16 A fuse. The final installation is shown in Figure 6.1 (left).

21 PowerMeters (see Section 4.3) have been deployed in the apartment. These have properly been connected to WiFi and it has been checked that the WiFi signal quality (*RSSI*) of each PowerMeter exceeds −60 dBm to be certain that data can be sent flawlessly. Some appliances like the *oven* and the *fume extractor* are directly connected to the mains. To measure those appliances, a special version of the PowerMeters with screw terminals has been used. Figure 6.1 (right) shows two PowerMeters connected to the *espresso machine* and *coffee grinder*.

Modern households can easily include more than 40 appliances (cf. Table 3.1). The apartment of FIRED includes 68 appliances in total. Many of these appliances are typically only used occasionally, connected to power on demand, and sometimes to a different socket than before. Therefore, connecting a continuously sensing meter to each appliance is unreasonable. Instead, appliances of the same category (e.g., routers) or devices which are only used simultaneously (e.g., monitor and PC) were connected to the same PowerMeter. Appliances which are only plugged in occasionally and typically not at the same time (e.g., mixer and vacuum cleaner)

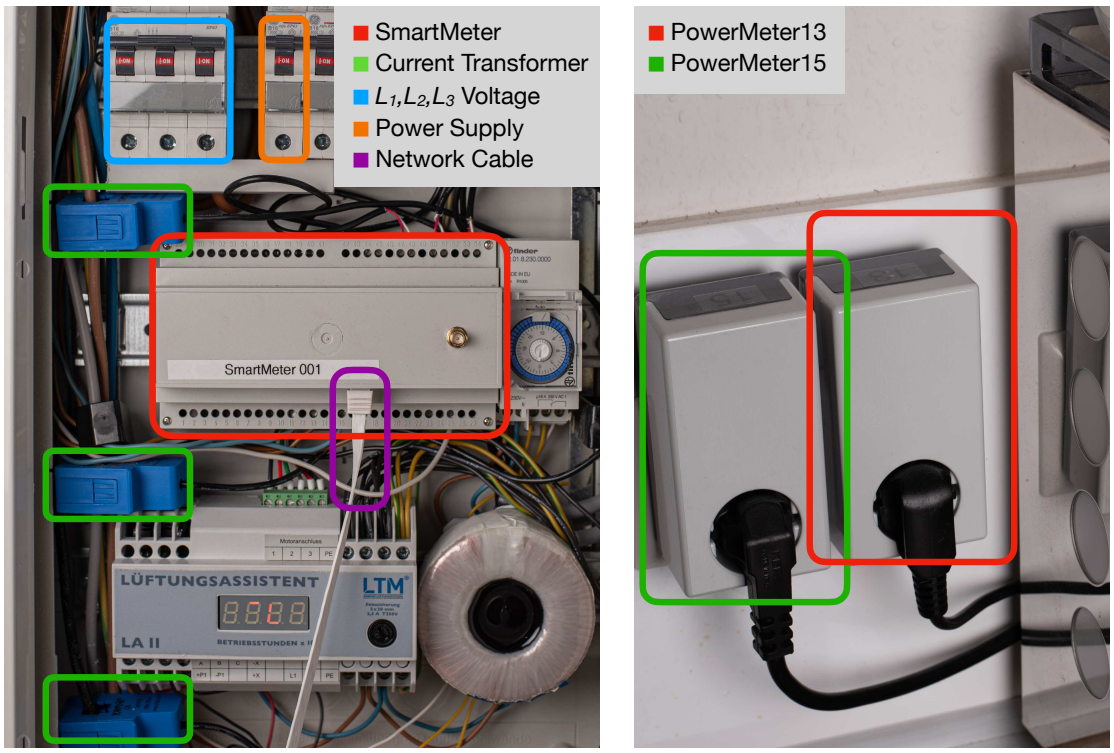

Figure 6.1: (Left) SmartMeter installed in the apartment's fuse box. (Right) PowerMeters with ID 13 and 15 connected to the *coffee grinder* and the *espresso machine*.

were connected to a dedicated PowerMeter (*powermeter11*). When an appliance was connected to or disconnected from this PowerMeter, a corresponding entry was manually added to a digital calendar which has been exported to a log file and is included in the dataset. This means that the connected appliance has changed over time, but the meter has continuously taken measurements.

Moreover, temperature, and humidity sensors were installed in the living room, bedroom, and child's room. As already mentioned, a ZigBee logger was set up to capture state changes of the apartment's lighting. Furthermore, a 433 MHz bridge was installed to capture state changes of the overhead kitchen light and an infrared bridge was installed to record key presses of the television's and HiFi system's remotes. The corresponding components have already been described in Section 4.4.

To properly connect all individual data acquisition devices to a central recording PC, the apartment was equipped with three additional WiFi access points. The power consumption of the recording PC and all access points have been recorded

individually. These also contribute to the apartment's aggregated consumption. The recording PC gathered the measurements of all electricity meters (SmartMeter and PowerMeters), all sensors, and all bridges. The PC stored the measurements into files frequently and pushed the files to a cloud server for persistent storage. For backup purposes, the local files on the recording PC were not deleted. The *recording manager* (cf. Section 4.6) running on the PC was set up to notify the home owners on any error or warning via email.

6.2 Data Records

The provided data include voltage and current measurements at high sampling rates taken from the aggregated mains signal and 21 individual outlets. To get a quick insight into the data, FIRED contains per-day and per-appliance summary files with derived active, reactive, and apparent power measurements. The root directory of the dataset contains folders with the *raw* and *summary* data. The data are stored as multiple Matroska container into sub-folders named *powermeter<ID>* and *smartmeter001*, respectively. File size and, therewith, file loading times are kept reasonable by splitting all files at regular time intervals. The local time of the first sample is appended to each filename in the format *<year>_<month>_<day>___ <hour>_<min>_<sec>*.

Table 6.1 shows the mapping of each recorded appliance to the used PowerMeter (ID). For more information about each appliance, its brand and model are shown. The power rating (P) according to the manufacturer of the appliance as well as the average ($\overline{P}$), and maximum (P_{max}) power observed during recording is further provided. Φ corresponds to the live wire (L_1, L_2, or L_3) to which the specific PowerMeter was connected to. A complete list of all appliances in the apartment, including the individually monitored appliances, is part of the dataset.

Furthermore, temperature and humidity sensor values of multiple rooms as well as state changes of many (smart) appliances are included. In addition, two weeks of data have been annotated with event labels. Sensor values and labels are stored as *.csv* files in *annotation* and *labels* folders, respectively.

Additional information about each electrical appliance installed in the apartment and not limited to those that have been individually monitored is included in the *info* folder of the dataset. This information includes the appliance brand, model number, and website links with further information.

Table 6.1: Appliances recorded via PowerMeters. *ID* represents the specific PowerMeter used for recording. For *PowerMeter11* the connected appliance changed during recording. P is the power according to the appliance manufacturer, Φ is the *gird line* the appliance is connected to (L_1, L_2, or L_3), P_{max} is the maximum average power drawn for the duration of one second, and $\overline{P}$ is the average power during the recording. The unit of all power measurements is Watt.

ID	Connected Appliance	Brand	Model	P	Φ	P_{max}	$\overline{P}$
08	Baby Heat Lamp	Reer	FeelWell	600	2	611.9	0.3
09	Fridge	IKEA	HUTTRA	1000	3	1138.8	18.0
10	Smartphone Charger #1	-	2 Port USB	10	3	12.7	1.7
11	Different Devices				3	1898.7	3.1
12	Smartphone Charger #2	-	4 Port USB	25	1	27.9	2.8
13	Coffee Grinder	Graef	Cm800	128	3	206.9	0.1
14	Smart Speaker	Apple	HomePod	15	3	3.6	0.2
15	Espresso Machine	Rocket	Appartamento	1200	3	1230.6	29.8
16	Kettle	Aigostar	Adam 30GOM	2200	3	1958.8	2.9
17	Hairdryer	Remington	D3190	2200	1	1934.9	1.0
18	Router #1	Apple	Airport Extreme A1521	10.3	1	27.9	19.3
	Router #2	Telekom	Speedport Smart 1	10			
	Telephone	Gigaset	A400	1			
19	Printer	EPSON	Stylus SX435W	15	1	21.5	0.2
20	Office PC	Apple	Mac Mini A1993	85	2	236.1	59.5
	27” Display	Apple	Thunderbolt display	200			
	Speaker	Logitech	Z2300	240			
	Smartphone Charger #3	Apple	MD813ZM/A	5			
	Access Point #2	Apple	Airport Express A1264	8			
21	Media PC	Apple	Mac Mini A1347	85	3	45.4	13.0
22	HiFi System	Onkyo	TX-SR507	160	3	85.9	15.3
	Subwoofer	Onkyo	SKW-501E	105			
23	Television	Samsung	UE48JU6450	64	3	151.0	12.3
24	Light+Driver	IKEA	-	40	3	36.6	1.8
25	Oven	IKEA	MIRAKULÖS	3480	3	2491.0	9.7
26	Access Point #3	Apple	Airport Express A1392	2.2	3	2.7	2.2
27	Router #3	Netgear	R6250	30	1	56.9	15.7
	Recording PC	Intel	NUC8v5PNK	60			
28	Fume Extractor	IKEA	WINDIG	250	3	249.3	1.1

6.2.1 Voltage and Current Data

All PowerMeters sampled the current and voltage waveforms at a rate of 2 kHz. In theory data can be sampled and sent at up to 7.875 kHz using a single PowerMeter. If 21 PowerMeters are used simultaneously, however, the available WiFi bandwidth

limits the amount of data that can be sent simultaneously by all meters in the same WiFi range. Therefore, a sampling rate of 2 kHz has been chosen as a trade-off between reliability and temporal data resolution (see Section 6.4.3 for more information).

The SmartMeter recorded voltage and current waveforms of L_1, L_2, and L_3 at a sampling rate of 8 kHz. The ADC installed in the SmartMeter allows to sample these waveforms with a maximum sampling frequency of 32 kHz, but, again, higher reliability was preferred over better time resolution. The lower sampling rate is, however, still in line with the findings of Armel et al. that "*there may be little additional benefit between 15 kHz to 40 kHz because of the noise in that range in real buildings*" [53]. Furthermore, a reduced sampling rate leads to smaller files and simplifies data handling for the user.

Each file contains 600 seconds of data stored into a single audio stream inside a multimedia container. For the aggregated data, each audio stream has six channels (*v_l1*, *i_l1*, *v_l2*, *i_l2*, *v_l3*, *i_l3*) representing the current and voltage waveforms for the three supply legs. The audio streams for the individual appliance data contain two channels (v, i). The number of samples in each file should match the time distance to the next file. If this is not the case, no data are available for that meter during the particular time period. This occurred during a reliability reset each day at midnight and rarely for single meters due to occasional data loss as depicted in Section 6.4.3.

Data conversion and calibration are not required, as both the SmartMeter and all PowerMeters have been calibrated in advance and calculate the physical quantities from the raw ADC samples (*Volt* for voltage and *Milliampere* for current measurements). No additional data pre-processing is applied. The provided voltage and current data can be seen in Figure 6.2. Plot 1-3 show data of the SmartMeter while plot 4 and 5 show the simultaneous measurements of two additional PowerMeters. The plots do not only highlight the high temporal resolution of the data but also the achieved clock synchronization. The rush-in current shown in the PowerMeter data (Figure 6.2 plot 4) matches the rush-in current seen in L_3 of the SmartMeter (Figure 6.2 plot 3). A time shift between the measurement devices of around 10 ms can be observed. Even after 16 hours of continuous recording, the offset between the SmartMeter and PowerMeters was still below one mains cycle, highlighting the effectiveness of the realized clock synchronization (see Section 6.4.4).

The current and voltage waveforms of the recording of *powermeter15* are mirrored around the x-axis. This can be seen, as the rush-in current is in the positive direction for the PowerMeter while it is negative for the SmartMeter. The mirroring is originated in the fact that this specific PowerMeter measured the neutral wire instead of L_3. At an ordinary outlet, one port is connected to the neutral wire

while the other is connected to an active wire. The connection orientation does not influence the appliance operation, however, it influences the relative ground to which current is measured. Hence, depending on the wiring of the outlets and the orientation of the PowerMeter, it either measures the voltage of and the current flowing in the outlets' neutral or live wire. Information about the specific connection of each PowerMeter is included in the dataset.

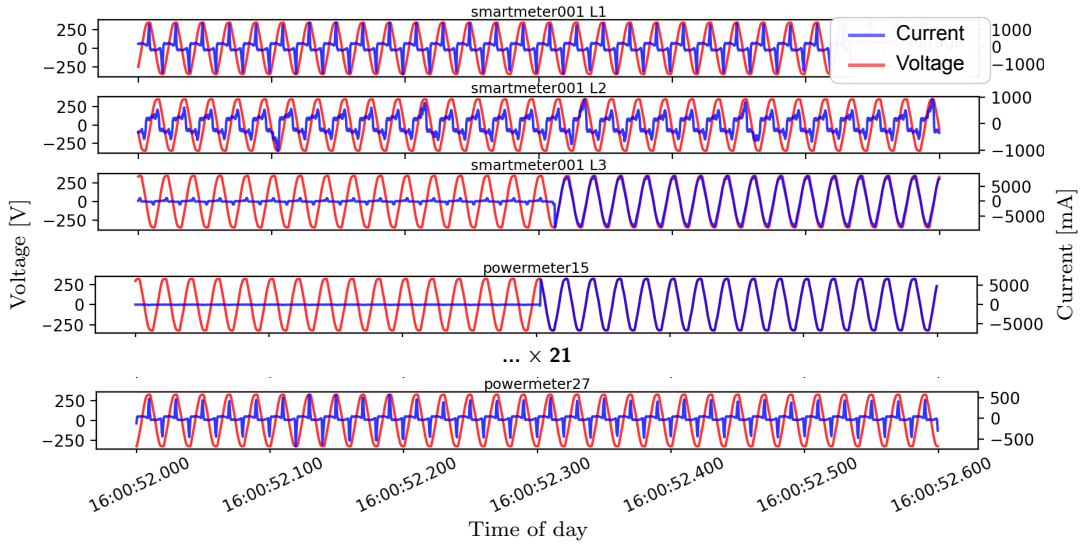

Figure 6.2: Voltage (red) and current (blue) waveforms of *smartmeter001*, *powermeter15*, and *powermeter27*. The recording was taken on June 9, 2020 at around 4 pm. The same appliance switch-on event of the *espresso machine* is visible in the recording of L3 of *smartmeter001* and of *powermeter15*.

6.2.2 Derived Power Data

The derived 50 Hz and 1 Hz active, reactive, and apparent power are stored for each meter in individual multimedia containers, and contain one day of data. Times for which the power could not be calculated since no voltage and current data were available, were marked with a power of constant zero to maintain an equidistant time period between samples. Constant zero plateaus can be identified easily as they do not represent valid measurements which are always non-zero due to omnipresent measurement noise. Figure 6.3 shows the single-day active power consumption of the apartment. The contributions of the six appliances which consumed the most power on this day are shown as individually colored blocks. The

power consumption of the remaining individually metered appliances are aggregated and plotted as the block *Others*. The aggregated power consumption of the SmartMeter (i.e., the sum of L_1, L_2, and L_3) is shown as the black line *mains*. Ideally, the superposition of the power of all individual meters should match the aggregated power. Nevertheless, a small margin can be observed in Figure 6.3. This gap is caused by hard-wired appliances such as the lighting and the ventilation system which are not monitored individually (see Section 6.4.2 for more information).

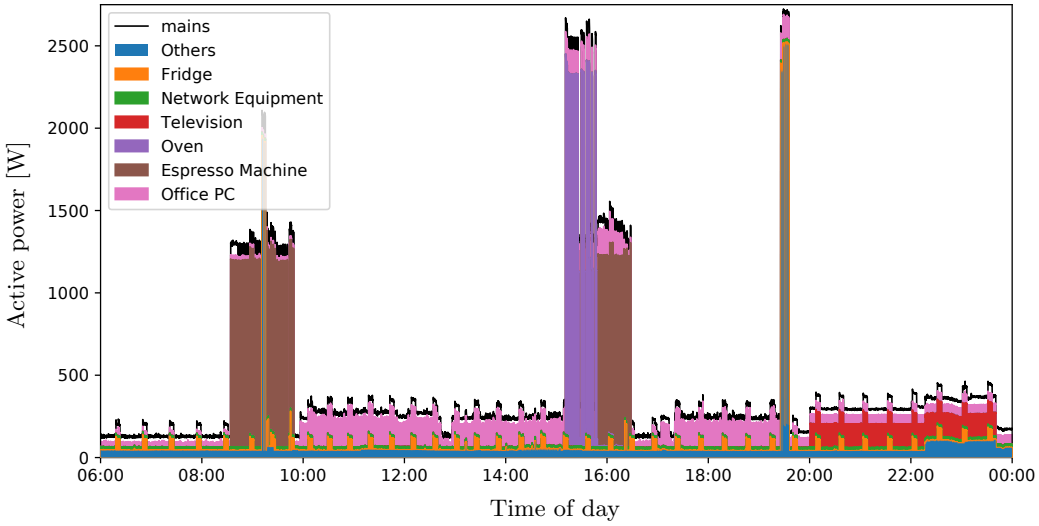

Figure 6.3: The power consumption of the apartment over one active day (July 2, 2020). The power signal is downsampled to one sample every 3 s. The black line indicates the power consumption recorded by the SmartMeter. The contribution of the six top-most consumers is shown as stacked colored blocks. The consumption of the remaining individually metered appliances are aggregated and shown as the blue block *Others*. A slight offset between the SmartMeter and the accumulated power of all PowerMeters can be seen.

6.2.3 Logs

The dataset's *annotation* folder contains 33 tab-separated *.csv* files. The first column of each file includes the timestamps associated with the events or sensor readings. These files can be divided into three categories:

Smart lights:
The state changes of each light bulb in the apartment is logged. The filenames of

these logs have the format:

light___<room>___<deviceName>___<deviceModel>.csv

room represents the name of the room the light is installed in, *deviceName* represents how this light is used (e.g., ceiling light), and *deviceModel* matches the specific model name of the light. The file's second column contains the states of the light (*on* or *off*), the third column contains the light's intensity levels (0 to 100 %), and the last column contains the light's *RGB* colors in hexadecimal. If setting different colors is not supported by the light, the column only shows *None* values. As individual measurements of the apartment's lighting have shown, the installed smart lights consume almost constant power linearly increasing with the light's intensity level. Since information of the lights' state and intensity setting is available for the complete recording duration, this information can be used to estimate the power consumption of each light individually. The smart light logs of two days are shown in Figure 6.4. The *hallway ceiling light* consists of three light bulbs and is triggered by a passive infrared sensor. Hence, all three light bulbs are turned on if a resident walks through the hallway which can be seen in the figure throughout the days.

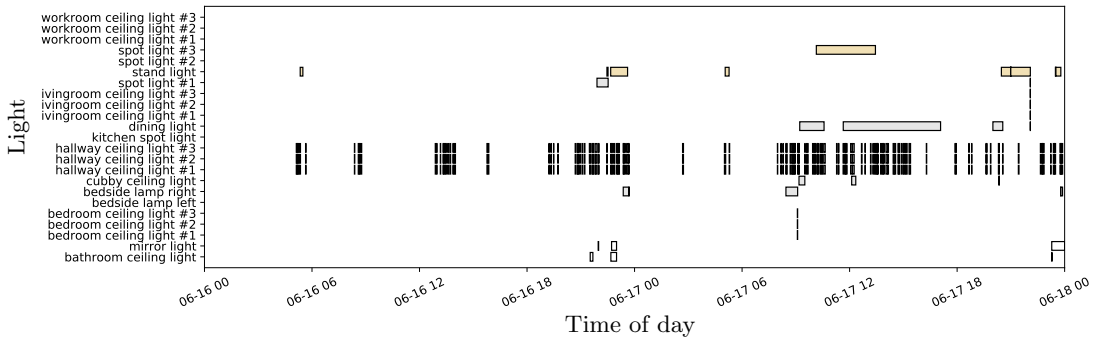

Figure 6.4: Two days of light usage information (starting at June 16, 2020). The time of day is shown on the x-axis while the particular light is listed on the y-axis. The black-framed boxes represent times when lights are active. Each box is filled according to the light's color and intensity.

Sensor readings:

The readings of temperature and humidity sensors are stored in files following the naming scheme:

sensor___<room>___<sensorType>.csv

sensorType is either *hum* or *temp* for humidity or temperature readings, respectively. Each file's second column contains the sensor readings. Temperature readings are stored in degrees Celsius and humidity readings in percent, respectively.

All values have floating point precision (32-bit). Samples are not acquired equidistant, as the sensors only send new values on a transition.

Device info:
Certain installed smart appliances or bridges allow to capture events of appliances in the apartment. Thereby, it is e.g., possible to recognize when a resident pressed a certain key of the television remote. Such events are logged in files with the following name format:

device__<room>__<deviceName>__<deviceModel>.csv

Each file's second column gives information about the current appliance state or the particular event. The file of the *HiFi system*, for instance, includes key-presses such as *power* or *vol_up* while the files of the *espresso machine* include the overall number of espressos made by the machine. When this number increased, the PowerMeter connected to the *espresso machine* recorded the electricity consumption required to make an espresso. Figure 6.5 shows the logs for the *television*, the *HiFi system*, and the *espresso machine*.

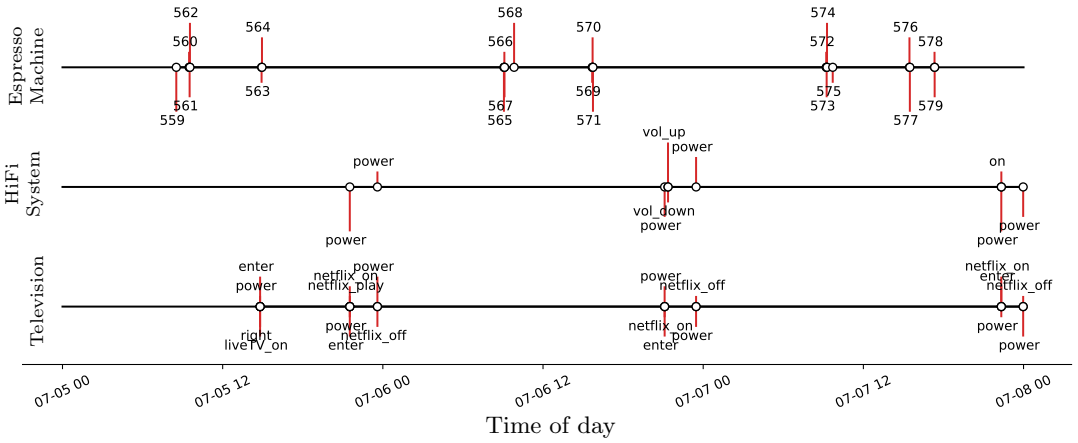

Figure 6.5: Three days of appliance logs (June 19 to 22, 2020). The data of the *espresso machine* show the numbers of espressos made, while the data of the *HiFi system* and *television* show key-presses on the remote.

6.2.4 Labels

The Annoticity labeling tool (see Section 5.2) was used to fully label all events that occurred within two weeks of the FIRED data (from July 22 to August 4, 2020). The tool generated an initial set of labels which was afterwards modified by visually inspecting the data. False events were removed, missing events were added, and a distinct and descriptive textual label was assigned to each event

representing the new state of the appliance. The labels were stored as CSV and are part of the dataset. Figure 6.6 shows the richness of these event labels for a time period of only two minutes. The figure illustrates the active and reactive power consumption of the *refrigerator* and the *espresso machine*. The aggregated consumption of supply leg L_3 is shown on top. The labels represent the state of the appliance and the black lines mark the timestamp of an event. The event timestamps are further highlighted in the aggregated consumption to demonstrate the synchronicity of the data streams.

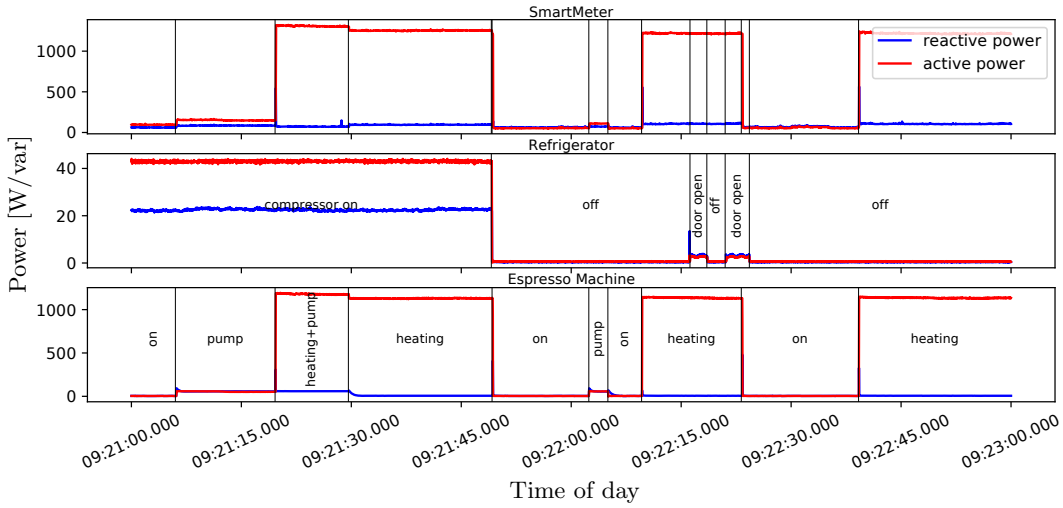

Figure 6.6: Event labels for two minutes of electricity data of the *refrigerator* and *espresso machine*. The aggregated consumption recorded with the SmartMeter is shown at the top with the position of all event labels marked.

6.3 Data Statistics

Overall, 53328 hours of raw current and voltage waveforms have been collected for the FIRED dataset. Figure 6.2 highlights the richness of the captured data for the SmartMeter and the PowerMeter units. Figure 6.3 shows the active power extracted for each individual appliance. It further emphasizes the contribution of each appliance to the total power consumption on this day. According to [166], the average consumption of a comparable three person household in Germany is 7.12 kW h per day. The analysis of the SmartMeter data of the FIRED dataset reveals an average electricity consumption of 6.06 kW h per day, which is slightly lower than the typical average consumption. This is, however, expected as the

data do not contain the electrical energy consumption of the washing machine, dryer, and freezer.

Figure 6.7 **a** shows the consumption of six appliances at the time of day averaged over the entire recording duration. This provides a good indication of usage behavior. For example the *espresso machine* shows two distinct peaks, one in the morning at around 9 am (morning coffee), and one in the afternoon at 3 pm (coffee break). Similarly, the *television* is mainly used after 8 pm. In comparison, the *router* does not show any significant peak. It can also be seen, that the *office PC* has a high standby consumption of around 35 W and is used mainly between 9 am and 6 pm. Figure 6.7 **b** shows the distribution of power demand for the same appliances. Some state information can already be derived from these plots. The *hairdryer* shows two distinct states corresponding to two different temperature settings. The *office PC* shows three peaks. The peak around 35 W represents the already mentioned standby consumption, the peak around 50 W represents the PC in its *on* state, and the 140 W peak includes the *on* state of the 27 inch monitor which is connected to the same PowerMeter. The *espresso machine* consumes a huge amount of power (1200 W) during its heating cycles but is mostly idle (5 W) in between. The consumption of the *router* and *kettle* resemble a Gaussian distribution centered around 19 W and 1890 W, respectively. It is assumed that the Gaussian of the *kettle* would be smoother and better visible if more samples would be available for its *on* state (as for the router).

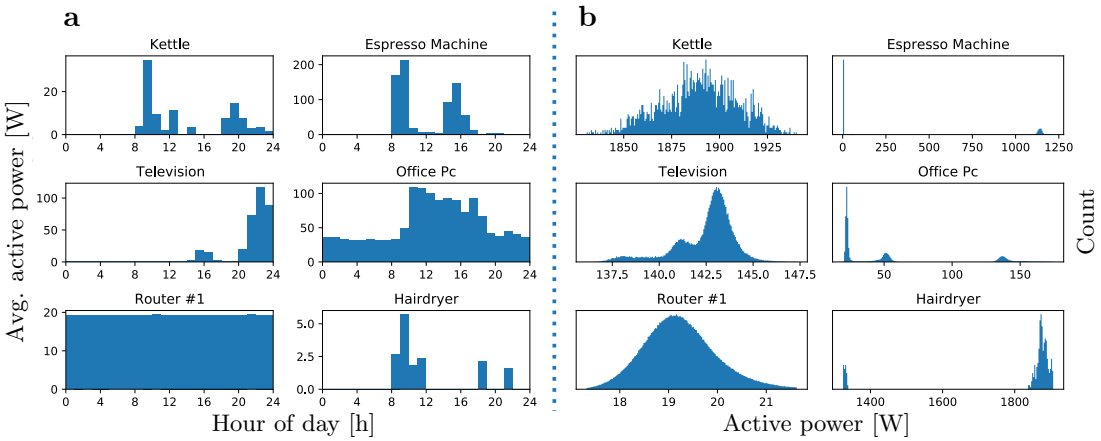

Figure 6.7: Appliance usage over the complete recording duration. (a) shows the daily usage patterns of the appliances with the consumed average power for the hour of the day. (b) shows the histogram of the power demands; a 2 W threshold was set to omit data in which no power is drawn.

6.4 Technical Validation

Measurements of the FIRED dataset are provided without applying any pre-processing or filtering except for the calibration and on-meter conversion to physical quantities. The recording framework is equipped with different mechanisms to cope with real-world effects such as network dropouts or clock drifts. The integrity of the acquired measurements is analyzed in the following.

6.4.1 Calibration

As already mentioned, each meter has been calibrated in advance using a dedicated electricity meter (*ENERGY-LOGGER 4000* by *VOLTCRAFT* [167]). According to its datasheet, the meter has a stated accuracy of 1 %. It was, therefore, used to determine the calibration parameters of each deployed SmartMeter and PowerMeter. Ten electrical loads with different power consumption ranging from 5 to 2000 W were used and a linear calibration was applied to each measured value x_{raw} (voltage and current) according to

$$x_{cal} = x_{offset} + m \cdot x_{raw} \,. \tag{6.1}$$

The calibration parameters m and x_{offset} for voltage and current measurements of each meter were stored permanently in its non-volatile memory. After about four months, the calibration has been repeated to see if aging affects have already invalidated the calibration. Such effects could not be observed.

6.4.2 Residual Power

Ideally, the sum of the electricity consumption of all individually metered appliances should be equal to the consumption recorded at the aggregated level according to Equation 2.2. However, a slight offset will be perceivable in any real dataset. This offset is referred to as the *residual power*. The residual power is the portion of the total consumed power which is not metered by an individual meter, i.e., the portion for which no ground truth data are available. One goal for the FIRED dataset was to minimize this portion in order to provide reliable ground truth data for supervised machine learning algorithms.

The residual power observed in the FIRED dataset (see Figure 6.3) is mainly due to non-monitored hard-wired appliances in the apartment such as the lighting and the ventilation system, but also due to the power consumption of the distributed PowerMeters. The individual consumption of each light bulb and, therewith, the

lighting can be estimated using the log files provided with the dataset. To demonstrate that this is feasible, power estimates have been generated using these log files and additional individual light recordings. The consumption of the remaining unmonitored appliances (including the consumption of 21 PowerMeters) is the base power consumption of the apartment. It can be estimated at times when lights are turned-off and the majority of appliances do not consume any power which is typically during the night or in case of absence of all residents. The base power P_{baseLx} of each individual supply leg $x \in \{1, 2, 3\}$ was calculated as

$$PM_{Lx} \coloneqq \{\, pm \in PM \mid \text{phase of } pm \text{ is } x \,\}, \tag{6.2}$$

$$L_{Lx} \coloneqq \{\, l \in Lights \mid \text{phase of } l \text{ is } x \,\}, \tag{6.3}$$

$$\mathbf{P}_{baseLx} = \mathbf{P}(SM_{Lx}) - \sum_{pm \in PM_{Lx}} \mathbf{P}(pm) - \sum_{l \in L_{Lx}} \mathbf{P}(l)\,. \tag{6.4}$$

SM_{Lx} represents the SmartMeter data of the grid line Lx, PM is the set of all PowerMeters, PM_{Lx} is the set of PowerMeters that are connected to live wire Lx, $Lights$ is the set of all lights, L_{Lx} is the set of lights connected to Lx, and $\mathbf{P}(y)$ is the extracted power trace of a meter or light y. It is assumed that the base power is normally distributed. Thus, all points in $\mathbf{P}_{baseLx}$ that are farther than σ from the mean value have been removed. Finally, the average P_{baseLx} is calculated from the cleaned signal.

Figure 6.8 shows the active power consumption including the lighting and the estimated base power with a remaining RMSE (see Equation 2.18) of 17 W. The residual power could not be completely eliminated. The reason has been identified to most likely be the apartment's ventilation system which reverses the direction of the air flow frequently.

6.4.3 Availability

Data are available for 99.96 % of the complete recording duration. 1405 minutes of data are missing, mainly due to a reliability reset which is performed at 12 am midnight. Occasionally, due to WiFi connection outages and an erroneous implementation of the TCP/IP stack on the ESP32 microcontroller, some data packets were lost. However, a packet only accounts for less than 20 ms of data. Once detected, the missing samples are replaced by zeros to maintain the correct timestamps for all remaining samples. It is still possible to identify these time periods as voltage and current zero plateaus cannot occur in natural situations.

The length and amount of the dropouts are illustrated for each meter in the boxplot shown in Figure 6.9. The figure shows that *powermeter14* and *powermeter22* had more dropouts compared to all other meter. This was identified to be originated

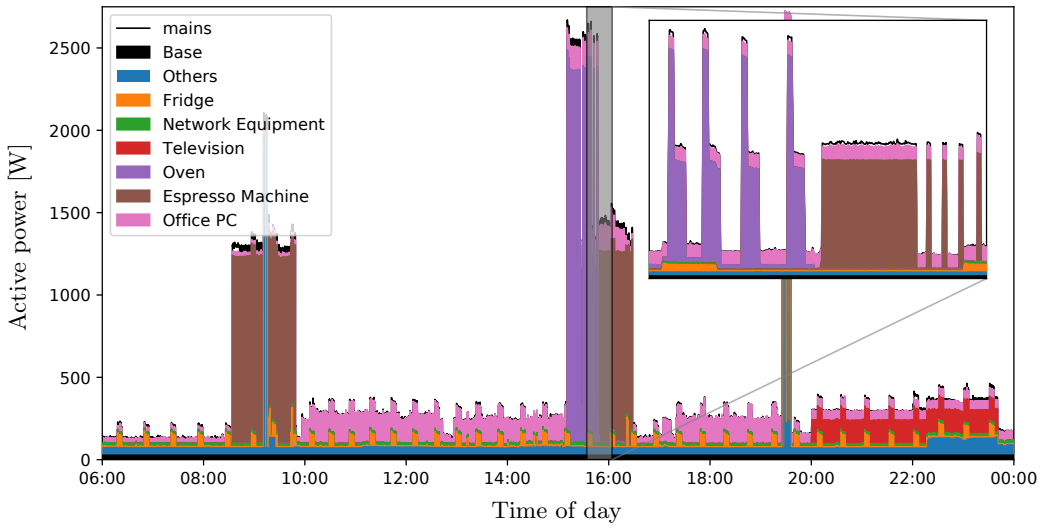

Figure 6.8: The power consumption of the apartment over one active day (July 2, 2020). The power is downsampled to one sample every 3 s. The black line indicates the power consumption recorded by the SmartMeter. The contribution of the six top-most consumers is shown as stacked colored blocks. The consumption of all remaining appliances and the reconstructed consumption of the apartment's lighting is aggregated and shown as the blue block *Others*. The black base block represents the apartment's base power which has been estimated as 26.66 W on average for this day.

from an unstable WiFi condition as the RSSI values reported by both meters were the lowest of all. In contrast, the SmartMeter which is connected over a reliable cable connection shows the least number of dropouts and the smallest durations for the dropouts.

Overall, the duration of dropouts is comparatively small compared to dropouts of multiple days such as present in the REDD dataset. While the average dropout duration fluctuates around the time it takes for a meter to reset at midnight (around eight to 30 seconds), longer dropouts occurred only rarely. Overall, over 3198275 minutes of high-frequency time-synchronized voltage and current waveforms are still available and the main time period in which data are missing is located each day at around 12 am. This represents a time period in which typically significantly less appliances are turned on.

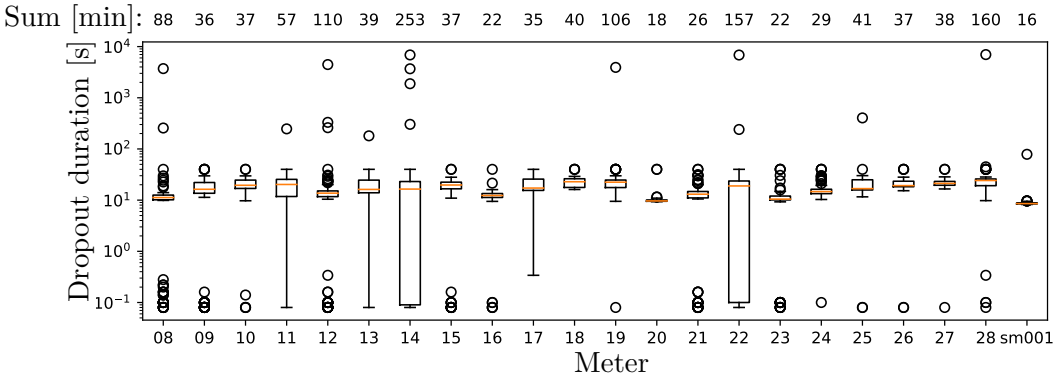

Figure 6.9: Semi-logarithmic boxplot showing the duration of the dropouts occurring on all meters during data acquisition. The mean value of each meter corresponds to the time of the reliability reset. Occasionally, shorter dropouts occurred. Longer dropouts are rare. The total dropout duration of each meter is displayed on top.

6.4.4 Clock Synchronization

The used clock synchronization technique has previously been detailed in Section 4.5. Figure 6.2 highlights the achieved clock synchronization in the FIRED dataset due to the methodologies explained in Section 4.5. The figure shows the voltage and current measurements of two PowerMeters and the SmartMeter. The current data of *powermeter15* shows a rapid rise in the current consumption due to a heating element in the connected *espresso machine*. The corresponding increase can be also observed in the measurements of L_3 of *smartmeter001*. Both signals are shifted by around 10 ms. This highlights the achieved clock synchronization, as all timestamps are derived relative from the first sample obtained. This initial sample was obtained around 16 hours earlier (due to the reliability reset at 12 am). Even after 16 hours of continuous recording, the time shift was still below one mains cycle, allowing to synchronize the voltage and current waveforms with sub-cycle precision.

6.5 Discussion

In the beginning of this chapter, desired characteristics (design goals) of electricity datasets, which can be used to evaluate a wide variety of electricity-related algorithms (especially event-based NILM), have been stated. These characteristics are closely related to the challenges which have been defined for the proposed

recording framework in Section 4.1. Based on this framework, the FIRED dataset has been recorded and labeled. It features 101 days of electricity measurements (C3) of a residential apartment in Germany. This is significantly longer than most existing high-frequency datasets such as REDD [17] or BLUED [94]. Aggregated-level data are available as 8 kHz voltage and current waveforms while individual appliance data are available at 2 kHz for 21 appliances (C1, C2). While the aggregated sampling rate is matched or even exceeded by other datasets, to the best of the authors knowledge, no other residential dataset features high-frequency individual appliance recordings. The data are further time-synchronized with an accuracy of around 10 ms (C5) and have a coverage of 99.96 % over the complete recording time period (C3). Other datasets such as REDD [17] or UK-DALE [75] show a significant amount of missing samples due to bad wireless communication. The framework additionally provides a 1 Hz and 50 Hz summary with derived active, reactive, and apparent power measurements. All data are stored in Matroska multimedia containers (C7) with included metadata information such as timestamps and measured quantities. Additional files are included in the dataset which provide information about the apartments lighting states, room temperature, and appliance operation states (C4). Event positions and state labels have been added for two weeks of the data in a semi-automatic way using the Annoticity labeling tool (C4). Except BLUED [94], no other dataset was found which includes such information. Thus, the dataset fully meets all of the above challenges compared to the existing datasets which have been found to only meet some of these (cf. Table 4.1). The dataset itself and the tools to process it are provided as open source (C7) to support the research of NILM algorithms.

Data and Code Availability

The FIRED dataset is available under the creative common licence. Further information on how to use and download the data can be found at `https://github.com/voelkerb/FIRED_dataset_helper`.

Personally, I think it is more important to focus on the development of electricity-related algorithms which either help to save energy or provide convenience to home owners, rather than on recording novel datasets or writing code to use existing datasets. Therefore, a *helper* module is provided to simplify interfacing with the dataset. The module is written in Python - the programming language most often used by the machine learning community. It allows to quickly load data of interest into memory without the need to explicitly extract and convert these data from individual files. An example showing how to plot two hours of 50 Hz power measurements of the *fridge* is shown in Listing 1.

```python
import helper as hp
import matplotlib.pyplot as plt
from datetime.datetime import fromtimestamp
# Set FIRED base folder to the location where you downloaded the dataset
hp.FIRED_BASE_FOLDER = "~/FIRED"

# load 2h of 50Hz power data of powermeter09 (Fridge)
startTs, stopTs = hp.getRecordingRange("2020.08.03 17:25:00", "2020.08.03 19:25:00")
fridge = hp.getMeterPower("powermeter09", 50, startTs, stopTs)

# Construct timestamps
dates = [fromtimestamp(fridge["timestamp"] + i/fridge["samplingrate"])
    for i in range(len(fridge["data"]))]
# Plot data
plt.plot(dates, fridge["data"]["p"], label="Active power")
plt.plot(dates, fridge["data"]["q"], label="Reactive power")
# Format plot
plt.gca().set(xlabel='Time of day', ylabel='Power [W/var]', title='Fridge')
plt.gcf().autofmt_xdate()
plt.show()
```

Listing 1: Example of loading and plotting two hours of power data of the *fridge*.

7 Feature and Classifier Study

In this chapter, several features and classifier found in literature are evaluated for their suitability to classify appliances. While works by Kahl et al. [117] and Sadeghianpourhamami et al. [129] already contain similar feature analysis, this evaluation results in a proposed feature set which has a small overall dimension and is feasible to be extracted and processed on resource constrained embedded systems. The set has further been evaluated on several classification algorithms, leading to F_1-scores of up to 98 % on average across four publicly available datasets. The chapter includes the following contributions of [N21]:

- An introduction of a naive event detector, of 27 handcrafted features for appliance classification, and of four classifiers (see Section 7.1).
- A standalone analysis of all features and classifiers (see Section 7.2).
- A method to determine the best combination of features for given data while keeping the feature dimensionality low (see Section 7.3).
- Using this method, several combination of features are evaluated and the feature set $[P, cos\,\Phi, TRI, WFA]$ is proposed (see Section 7.3).

7.1 Preliminaries

This section details the event detection algorithm as well as the features and basic classifiers used throughout this chapter. It further states the used method and parameter sets for Hyperparameter Optimization (HPO), and details the Cross Validation (CV) procedure as well as the employed metrics.

7.1.1 Event Detection

While the event detector introduced in Section 5.1.1 performs quite well, it is computationally expensive due to windowing and the use of non-linear functions such as the logarithm. For the following evaluation a computationally fairly simple expert heuristic event detector was used to generate the ground truth and training

data. The detector is based on work by Weiss et al. [18]. For a comparison of different event detection methods, the reader is referred to Section 3.3.

The used expert heuristic utilizes a threshold-based algorithm applied on the apparent power signal (S). At first, the signal is filtered using the combination of a median and a mean filter to remove outliers and further smooth the signal. Both filters have a window size of 3 s. Afterwards, the differences between adjacent samples of the apparent power signal are calculated (ΔS). Next, a 3 VA filter is applied to the signal which sets all values below 3 VA to zero as

$$\Delta S_{filtered}[i] = \begin{cases} 0 & \text{if } \Delta S[i] < 3\,\text{VA}, \\ \Delta S[i] & \text{else}\,. \end{cases} \tag{7.1}$$

Each non-zero region in the filtered signal ($\Delta S_{filtered}$) is regarded as an event (either *up* or *down*). If multiple events happen within a time window of 3 s, only the first event is kept. This ensures that fluctuations after an event are not regarded as a new event. Figure 7.1 shows the different stages of the event detection process for the apparent power signal of an espresso machine. All significant events are clearly visible as peaks after the filtering process (green signal). The red and blue triangles mark the timestamps and the corresponding apparent power samples of the finally extracted events.

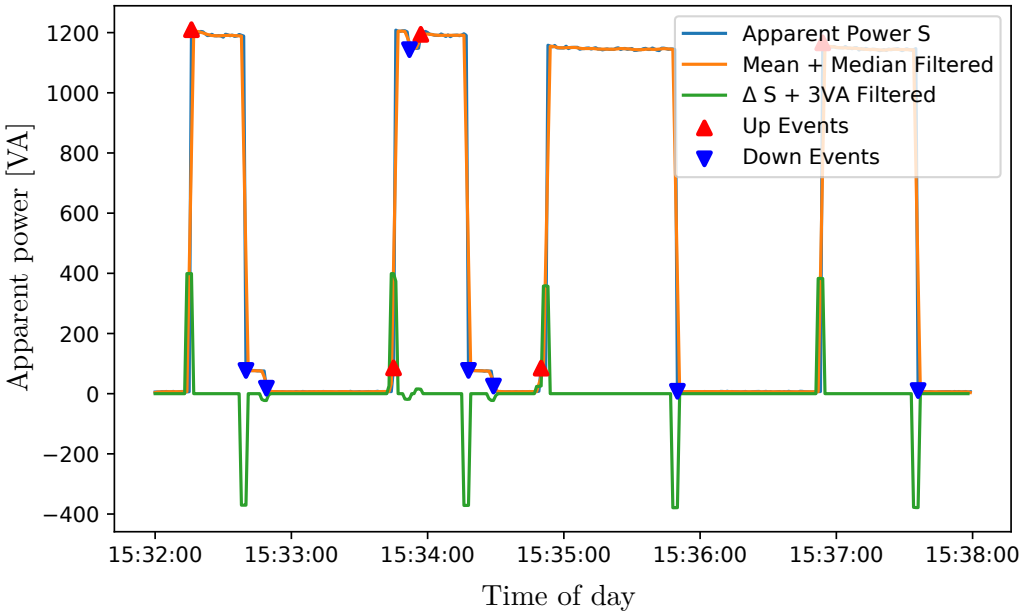

Figure 7.1: Event detection applied to an excerpt of the 1 Hz apparent power signal of the *espresso machine* (FIRED). The green signal remains after filtering and allows to extract up and down events.

To be able to calculate high-frequency features for a detected event, voltage and current waveforms 500 ms prior till 1 s after the timestamp of the event are extracted. This 1.5 s time interval is referred to as the Region of Interest (ROI) in the following. Furthermore, each ROI is forced to begin with the preceding positive zero-crossing of the voltage measurements, slightly shifting the event position. All 27 features explained in the following can be extracted from the ROI data corresponding to an event. For the feature and classifier analysis in Section 7.2 and Section 7.3, only start-up events taken from individual appliance recordings are used. This means that the current in the first 500 ms is close to zero and only contains measurement noise. If individual appliance data apart from start-up events, or aggregated-level data are used (as in Section 8.2 and 8.3 of Chapter 8), the current in the first 500 ms does not have to be zero and the overall signal contains contributions of other active appliances (cf. Kirchhoff's first law [29]). However, several of the extracted features meet the feature additive criterion [115, 128] which is based on Kirchhoff's first law. Therefore, the features can be cleaned using data of the first 500 ms (see Section 8.2 for more information). Figure 7.2 shows the current signal of the ROI for a start-up event of the fridge (cooling cycle) included in the FIRED dataset.

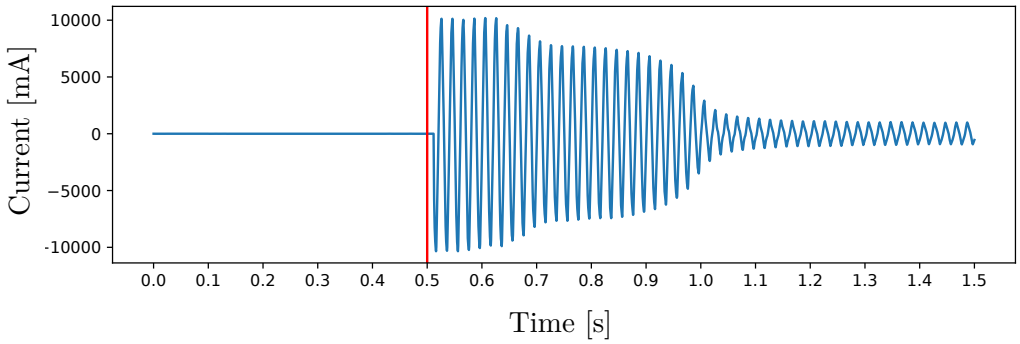

Figure 7.2: Example of the ROI current data of a fridge event in the FIRED dataset. The exact event position is marked in red.

7.1.2 Feature Selection

The following evaluation analyzes a set of 27 features which have been introduced by various domain experts in related works [117, 128, 129] for the task of appliance classification. All used features are summarized in Table 7.3 and can be extracted from the time or frequency domain of the ROI of an event. According to the Nyquist-Shannon theorem (see Equation 2.6), current and voltage waveforms with a sampling rate (f_s) of more than $2 \cdot (18 + 1) \cdot f_0$ are required because frequency

components up to the 18th harmonic (f_{18}) of the grid line frequency (f_0) are analyzed. Therewith, the sampling rate has to be higher than 1900 Hz for a European grid line frequency of 50 Hz. To avoid aliasing artifacts, a *Butterworth* low-pass filter (order = 6, $f_{\text{cutoff}} = 1\,\text{kHz}$) is applied to the current and voltage waveforms in order to suppress higher frequencies before any feature is extracted.

The feature set includes both transient and steady-state features. Steady state features include several electrical measurands such as the phase angle between voltage and current ($cos\,\Phi$), resistance (R), admittance (Y) as well as active (P), reactive (Q), and apparent power (S) which can be calculated on the basis of a single mains cycle. The corresponding formulas have already been introduced in Section 2.3.1. Transient features such as Current Over Time (COT) or Temporal Centroid (TC) describe the change of certain electrical characteristics (such as the current) over a specified time window. The set further includes steady-state and transient features proposed by several researchers which stem from extensive feature engineering such as the steady-state feature V-I Trajectory (VIT).

Before each individual feature is explained, the short hands for the current waveform $I_{WF}[j]$ and the RMS current $I_{RMS}[j]$ of main cycle j are defined as

$$I_{WF}[j] = [\textit{current samples of period } j]\ , \tag{7.2}$$

$$I_{RMS}[j] = rms\left(I_{WF}[j]\right)\ . \tag{7.3}$$

As mentioned earlier, the ROI for a given event is extracted from 500 ms before to 1 s after the event. To simplify the following formulas, I_{WF} is referred to as the current waveform from the event to the end of the ROI ($0.5\,\text{s} \leq t < 1.5\,\text{s}$). Similarly, $I_{WF}[j]$ and $I_{RMS}[j]$ with $j \in [0, \ldots, N]$ refers to the jth period after the event. For example, at $f_0 = 50\,\text{Hz}$, $I_{WF}[0]$ is the current waveform $i[t]$ with $500\,\text{ms} \leq t < 520\,\text{ms}$ from the ROI. The voltage waveform $U_{WF}[j]$ and the RMS voltage $U_{RMS}[j]$ of main cycle j are calculated alongside.

The **Crest Factor (CF)** shows the ratio of the peak value to the effective RMS value of the current and is an indicator for the sharpness of a transient or peak in the signal.

$$CF = \frac{max\left(|I_{WF}|\right)}{rms(I_{WF})} \tag{7.4}$$

The **Form Factor (FF)** is the ratio of the effective RMS current value to the mean of the absolute current values. As the RMS current is calculated from the squared current values, spikes have more influence on RMS than on the corresponding mean value. Hence, the FF can be used to distinguish linear loads from appliances that show larger spikes in the current signal such as SMPS-powered appliances.

$$FF = \frac{rms(I_{WF})}{mean\left(|I_{WF}|\right)} \tag{7.5}$$

The **Log Attack Time (LAT)** measures the logarithmic time until the current reaches its maximum value. Compared to other works (e.g., [117]), the maximum value refers to the maximum RMS current value over one main period. The LAT results in larger values for appliances for which the current rises slowly.

$$\begin{aligned} LAT &= 20 \cdot log_{10}\left(t_{maxCurrent}\right) \ , \\ t_{maxCurrent} &= j \cdot \tfrac{f_s}{f_0} \ . \end{aligned} \tag{7.6}$$

j is the main cycle at which $I_{RMS}[j]$ reached the maximum value.

Similarly, the **Temporal Centroid (TC)** measures the temporal balancing point of a signal's energy over a certain time period. If an appliance has a strong start-up current followed by a settlement of the current (e.g., a refrigerator due to start-up current of the built-in compressor) the corresponding TC value is lower compared to the value of an appliance for which the current starts to rise slowly.

$$TC = \frac{1}{f_0} \cdot \frac{\sum_{j=0}^{N}\left((j+1) \cdot I_{RMS}[j]\right)}{\sum_{j=0}^{N} I_{RMS}[j]} \tag{7.7}$$

The **Positive-Negative half cycle Ratio (PNR)** describes the ratio between the equivalent DC power of the positive part of a current mains period compared to its negative part. As the ratio depends on the direction the appliance is plugged in (i.e., which of the plug's connections is coupled with the active and which to the neutral wire), the value is inverted if the ratio is smaller than 1. Appliances which include e.g., dimmers or speed controllers show different behavior in the positive compared to the negative half-cycle.

$$PNR = \begin{cases} \frac{I_{RMS}(pos)}{I_{RMS}(neg)} & \text{if } I_{RMS}(pos) \geq I_{RMS}(neg) \\ \frac{I_{RMS}(neg)}{I_{RMS}(pos)} & \text{else} \end{cases} \tag{7.8}$$

Similar symmetry properties can be compared by the **Max-Min Ratio (MAMI)**, which relates the maximum current to the minimum current. For the same reason as stated for the PNR, the value is inverted if the ratio is larger than 1.

$$MAMI = \begin{cases} \frac{|min(I_{WF})|}{|max(I_{WF})|} & \text{if } \left|max\left(I_{WF}\right)\right| \geq \left|min\left(I_{WF}\right)\right| \\ \frac{|max(I_{WF})|}{|min(I_{WF})|} & \text{else} \end{cases} \tag{7.9}$$

The **Peak-Mean Ratio (PMR)** relates the maximum peak current to the mean of the absolute current value. Therefore, the feature can be used to differentiate loads with high current peaks (e.g., refrigerators) from linear loads (e.g., kettles).

$$PMR = \frac{max\left(|I_{WF}|\right)}{mean\left(|I_{WF}|\right)} \tag{7.10}$$

Likewise, the **Max-Inrush Ratio (MIR)** relates the RMS current of the first mains period including the event to the maximum absolute current in this mains period. This helps to recognize SMPS-powered appliances such as notebook power supplies where a high current peak can be observed when they are plugged-in and the embedded capacitors are energized.

$$MIR = \frac{I_{RMS}[0]}{max\left(|I_{WF}[0]|\right)} \tag{7.11}$$

The **Mean-Variance Ratio (MVR)** relates the mean of the absolute current to its variance, delivering an indication of the signal's steadiness which varies across appliances. Appliances with current spikes have a comparatively low MVR while linear loads tend to have a higher MVR due to a smaller variance.

$$MVR = \frac{mean\left(|I_{WF}|\right)}{var\left(|I_{WF}|\right)} \tag{7.12}$$

The **Waveform Distortion (WFD)** is the summed sample-wise difference between the current signal and a perfect sine wave with the same energy (Y). A pure resistive load will have a theoretical value of 0 as the sines will perfectly match.

$$WFD = sum\left(|Y| - |\boldsymbol{mean}\left(I_{WF}[0], \ldots, I_{WF}[9]\right)|\right) \tag{7.13}$$

The output of $\boldsymbol{mean}\left(I_{WF}[0], \ldots, I_{WF}[9]\right)$ is the average current waveform of the ten given mains cycles.

The feature **Current Over Time (COT)** describes the amount of RMS current in the first 25 consecutive mains cycles after an event. The mains cycle in which the event happens is not included, as its corresponding RMS current depends on the specific time the event occurred within the cycle.

$$COT = [I_{RMS}[1],\ I_{RMS}[2],\ \ldots,\ I_{RMS}[25]] \tag{7.14}$$

Figure 7.3 shows the current signal (ROI) of two appliances from the PLAID [23] dataset and the extracted COT.

To compress the COT into a scalar feature, the **Periods to Steady State (PSS)** indicates the first mains cycle after which a steady state is achieved. Most appliances such as the fridge shown in Figure 7.3 have a higher inrush current before a steady state is achieved. The time it takes for a steady state to settle can be exploited as a unique fingerprint of the appliance. The steady state is defined as a threshold (L) computed from the COT. PSS marks the first period after the event for which the current falls below this threshold as

$$\begin{aligned} L &= \tfrac{1}{8} \cdot \left(max\left(COT\right) - median\left(COT\right)\right) + median\left(COT\right)\ , \\ PSS &= \textit{first period } j \textit{ for which } I_{RMS}[j] < L\ . \end{aligned} \tag{7.15}$$

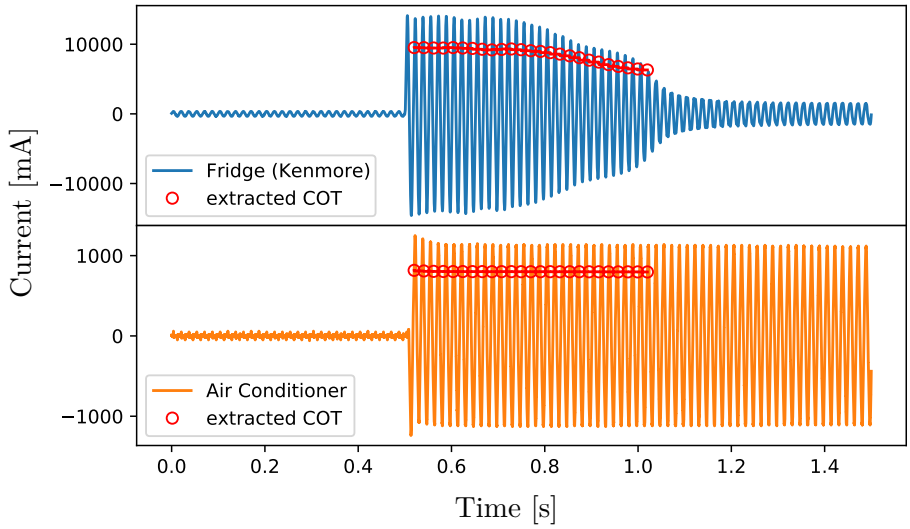

Figure 7.3: Start-up transients showing the ROI of a fridge and an air conditioner. Both signals are extracted from the PLAID [23] dataset. The red circles show the COT feature.

The **Inrush Current Ratio (ICR)** relates the current of the first mains cycle to the current of the last mains cycle. This allows to recognize appliances with a high inrush current e.g., appliances powered by SMPSs.

$$ICR = \frac{I_{RMS}[0]}{I_{RMS}[N]} \tag{7.16}$$

The **WaveForm Approximation (WFA)** indicates the overall waveform shape of the averaged mains periods of the current signal as

$$WFA = downsample\left(\boldsymbol{mean}\left(I_{WF}[0], \ldots, I_{WF}[9]\right), 20\right) . \tag{7.17}$$

Similar to the WFD feature, an average current waveform of the first ten mains cycles $I_{WF}[i]$ is calculated. Afterwards, the average waveform is downsampled to 20 samples using subsampling (i.e., use each $n = \lfloor f_s/(f_l \cdot 20) \rceil$ sample). Figure 7.4 shows the WFA of six different appliances from the WHITED [22] dataset. The waveform characteristics of the different appliances (blue) can be clearly seen. While the *hairdryer* and *lightbulb* show an almost pure resistive load behavior (cf. Figure 2.4), the *microwave* and *shredder* show a more distorted waveform with an imbalance of the positive and negative half cycle as well as a phase shift. The red dots indicate the downsampled WFA feature vector.

The **V-I Trajectory (VIT)** was first introduced by Lam et al. [121] in 2007. The authors state that it has a high discriminative power which has been proven by

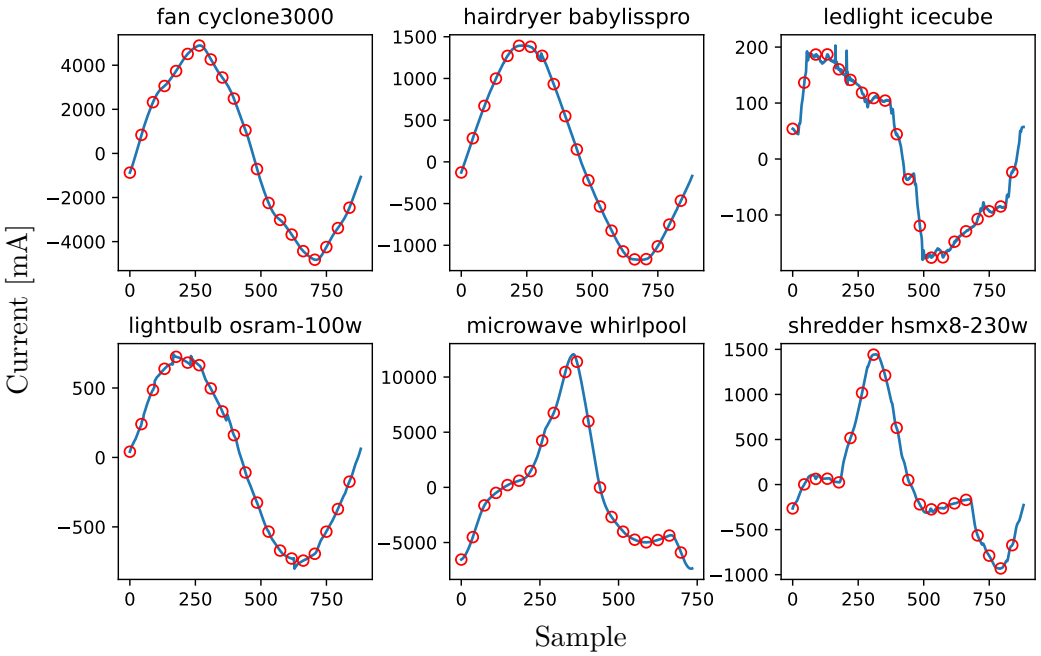

Figure 7.4: WFA of six different appliances from the WHITED [22] dataset. The red dots show the subsampled values of the feature vector.

other researchers such as [117, 122, 123]. To calculate the VIT the first ten mains periods of the current and voltage waveforms are averaged, similar as for the WFA feature. To create a deformation path, both signals are normalized, subsampled to 20 samples, and plotted against each other. This results in a feature vector of size 40. The average current $\widetilde{X}_{avg} = \widetilde{I}_{avg}$ and voltage $\widetilde{X}_{avg} = \widetilde{U}_{avg}$ waveforms are calculated as

$$X_{avg} = \boldsymbol{mean}\left(X_{WF}[0], \ldots, X_{WF}[9]\right), \tag{7.18}$$

$$\widetilde{X}_{avg} = downsample\left(\frac{X_{avg}}{max\left(|X_{avg}|\right)}, 20\right), \tag{7.19}$$

$$VIT = \left[\widetilde{I}_{avg}, \widetilde{U}_{avg}\right]. \tag{7.20}$$

Figure 7.5 shows the VIT of six different appliances from the FIRED dataset. While it can be assumed that most appliances can be distinguished quite well (e.g., *television*, *fridge*, *vacuum cleaner*, and *smartphone charger*), some appliances such as the *espresso machine* and the *kettle* may be difficult to keep apart using VIT as the exclusive feature.

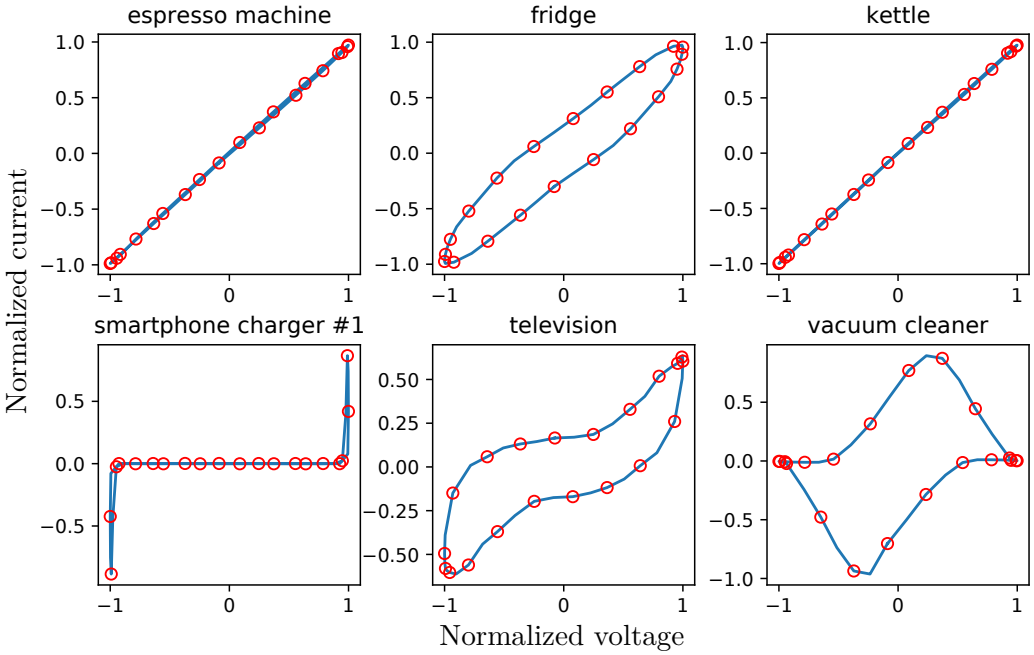

Figure 7.5: Averaged and normalized VIT of six different appliances from the FIRED dataset. The red dots show the subsampled values of the feature vector.

In addition to the already presented time domain features, several frequency domain features can be used for appliance classification. To transform a discrete time signal $x[n]$ of length N into the frequency domain, a discrete-time Fourier Transformation (DTFT) is used as

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j\left(\frac{2\pi n}{N}\right)k} \quad (k = 0, \ldots, N-1) \ . \tag{7.21}$$

The frequency corresponding to bin k depends on the sampling frequency of signal x and the number of sampling points N as

$$f(k) = k \cdot \frac{f_s}{N} \, . \tag{7.22}$$

For a simplified notation, X_f denotes the magnitude of the spectrum at frequency f. Furthermore, f_0 denotes the fundamental frequency of a signal and f_i the ith harmonic of f_0.

The **Total Harmonic Distortion (THD)** measures the logarithmic weighted relation of the sum of the first five harmonic components of the current signal

relative to the component of the fundamental frequency X_{f_0}. Strong harmonics in the current signal indicate a distorted waveform. Appliances which exhibit such characteristics are typically powered by SMPSs.

$$THD = 10 \cdot log_{10}\left(\frac{1}{X_{f_0}} \cdot \sum_{i=1}^{5} X_{f_i}\right) \tag{7.23}$$

The **Spectral Flatness (SPF)** (also known as the *Wiener Entropy*) is the ratio of the geometric mean to the arithmetic mean of the energy spectrum of the current signal. It is calculated as

$$SPF = \frac{\sqrt[N]{\prod_{k=0}^{N-1} X[k]}}{\frac{1}{N}\sum_{k=0}^{N-1} X[k]}\,. \tag{7.24}$$

As the geometric mean is always smaller or equal to the arithmetic mean, the value is $\in [0, 1]$. Pure resistive loads (e.g., kettles) induce little to no harmonics in the frequency spectrum. Hence, their corresponding geometric mean and, therefore, also the SPF will be close to 0. Appliances which induce larger harmonics (e.g., appliances powered by SMPSs) result in a larger value of SPF.

The **Odd-Even Harmonics Ratio (OER)** represents the ratio of the even harmonics of a signal to its odd harmonics. Depending on the type of SMPS or rectifier (half-wave or full-wave) used in an appliance, the current signal can have a characteristic imbalance of odd and even harmonics. The OER is calculated for the first 18 harmonics as

$$OER = \frac{mean\left(X_{f_1}, X_{f_3}, \ldots, X_{f_{17}}\right)}{mean\left(X_{f_2}, X_{f_4}, \ldots, X_{f_{18}}\right)}\,. \tag{7.25}$$

Similar to the TC in the time domain, the **Spectral Centroid (SC)** defines the balancing point of the spectrum. Appliances which use an SMPS typically have a higher balancing point because they induce higher order harmonics.

$$SC = \frac{\sum_{k=0}^{N-1} X[k] \cdot f(k)}{\sum_{k=0}^{N-1} X[k]} \tag{7.26}$$

The **Tristimulus (TRI)** compresses the lower, medium, and higher harmonics into a three dimensional feature vector. It, therewith, represents a compressed form of the HED feature explained in the following.

$$\begin{gathered} T_1 = \frac{X_{f_1}}{\sum_{k=0}^{N-1} X[k]}\,, \quad T_2 = \frac{X_{f_2} + X_{f_3} + X_{f_4}}{\sum_{k=0}^{N-1} X[k]}\,, \\ T_3 = \frac{X_{f_5} + X_{f_6} + \ldots + X_{f_{10}}}{\sum_{k=0}^{N-1} X[k]}\,, \\ TRI = [T_1, T_2, T_3]\,. \end{gathered} \tag{7.27}$$

The **Harmonic Energy Distribution (HED)** is a vector containing the first 18 harmonic current components normalized by the magnitude of the fundamental frequency as

$$HED = \frac{1}{X_{f_0}} \cdot [X_{f_1}, X_{f_2}, \ldots, X_{f_{18}}] \, . \tag{7.28}$$

Figure 7.6 shows the normalized spectrum of two appliances with a strong odd-even harmonic imbalance from the BLOND [21] dataset. The extracted HED is marked with red circles.

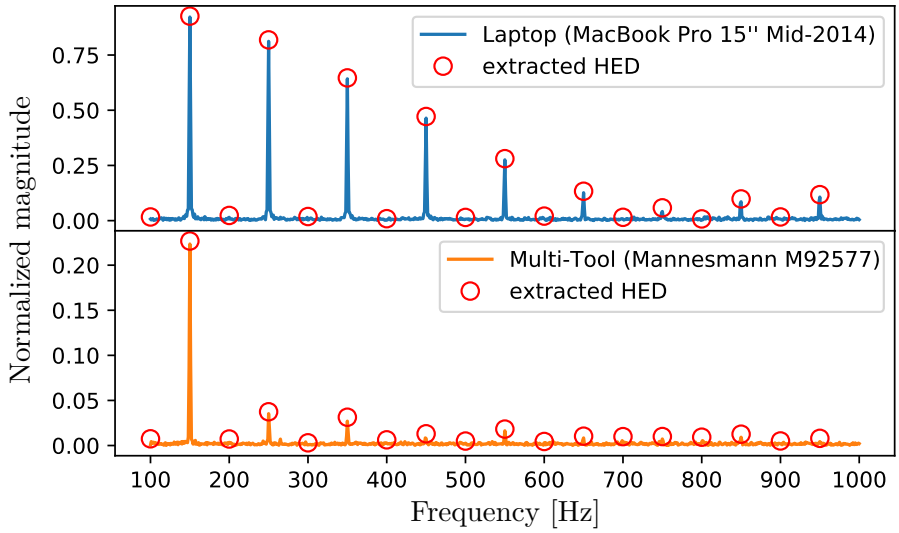

Figure 7.6: The spectra of a notebook and a rotary multi-tool (BLOND [21]), normalized by the magnitude of the fundamental frequency f_0. The extracted HED is highlighted with red circles.

Aliasing effects caused by sampling rates lower than $2 \cdot f_{18}$ would distort the spectrum and, thus, the extracted HED feature. Even if the sampling rate is chosen high enough, harmonics with a frequency larger than f_s are mirrored at the Nyquist frequency (cf. Equation 2.5) and may, therefore, still be visible. As shown in the top plot of Figure 7.6, the higher order harmonics ($>800\,\text{Hz}$) are exaggerated due to such interference. This behavior depends on the signal's sampling frequency and fundamental frequency. Proper low-pass filtering (with $f_{cutoff} \approx f_{18}$) and downsampling are applied to avoid such phenomena.

The evaluation carried out in Section 7.3 uses a combination of the presented features. As the features map different physical quantities and are of different ranges, feature scaling is applied to prevent undesired feature weighting. Each dimension x in the feature space is scaled using z-score normalization by $x_{\text{scaled}} = (x - \mu)/\sigma$ with the mean of all training samples μ and the standard deviation σ.

7.1.3 Classifiers

In the evaluations the following four classifiers are examined regarding their suitability for appliance classification: (1) SVM, (2) kNN, (3) RF, and (4) XGBoost. These classifiers have been pre-selected based on the following reasons: As the number of training samples, i.e., appliance events, is comparably low (cf. Table 3.2), classifiers are selected which generally work well on smaller training sets ($< 50k$ samples) compared to e.g., ANN which unfold their advantages especially if the training sets contain millions of samples. Furthermore, the number of events varies depending on the appliance type (e.g., more fridge events than iron events) resulting in an imbalanced training set. kNN is generally invariant to imbalanced data and RF, SVM, and XGBoost can be adapted to such training sets using class weighting or resampling strategies (*over-* or *undersampling*). Moreover, all selected classifiers can be easily adapted to multi-class classification tasks and, due to their low hyperparameter space, allow a comparably fast retraining including HPO. These are important properties for adaptive training which is required for any supervised NILM system to actually be deployed in a home (see Section 8.3).

To tune the parameters of each selected classifier (HPO), grid search is applied with a macro F_1-loss (see Section 7.1.4) based on the parameter sets listed in Table 7.1.

Table 7.1: Hyper-parameter grid used while tuning each classifier. *Comb.* shows the number of possible different combinations which need to be analyzed during grid search.

Classifier	Parameters	Comb.
kNN	$k \in [1, 2, \ldots, 20]$	20
SVM	$C \in [0.01, 0.1, 1, 10, 100, 1000]$ $\gamma \in [10000, 1000, 100, 10, 1, 0.1, 0.01]$ $kernel \in [RBF, linear]$	84
RF	$max_{depth} \in [10, 20, \ldots, 100]$ $n_{estimators} \in [10, 50, 100, 1000]$	40
XGBoost	$\gamma \in [0.5, 1, 1.5]$ $n_{estimators} \in [100, 200, 1000]$	9

For all remaining hyperparameters, the standard values of the *scikit-learn* [168] library are used.

7.1.4 Evaluation Setup and Metrics

For the following evaluations all features and classifiers introduced in Section 7.1.2 and Section 7.1.3 are used. As training and test data, start-up events from the publicly available datasets WHITED [22], PLAID [23], BLOND-50 [21], and FIRED (see Chapter 6) are extracted. It is noted that only appliance-level data are used, as the main focus of this evaluation is to find appropriate features and classifiers for appliance recognition. Information about the number of extracted events, the number of appliances, and the sampling rate of each dataset are summarized in Table 7.2.

Table 7.2: Information about the data extracted for the evaluation from the four used datasets. WHITED and PLAID include isolated measurements from a laboratory environment. BLOND-50 and FIRED include real-world individual appliance measurements taken at an office building and an apartment, respectively.

Dataset	Appliances	Appliance types	Sampling Rate	Start-up events
WHITED	119	54	44 kHz	1185
PLAID	202	11	30 kHz	1314
BLOND-50	26	8	6.4 kHz	8202
FIRED	20	20	2 kHz	5956

The WHITED dataset includes 119 different appliances from 54 appliance types, resulting in a comparably large number of classes to discriminate. The PLAID dataset includes 202 appliances from eleven appliance types and is, therefore, the dataset with the highest inner-class diversity. From the BLOND dataset, events are extracted from data between October 1, 2016 and January 20, 2017. In this time period, 26 different appliances from eight different appliance types were found to show distinct events. From the FIRED dataset, events from 20 individual appliances (only appliances which trigger events and not the recording hardware) are extracted from June 20 to September 22, 2020. While WHITED and PLAID already contain appliance start-up events, these must be extracted from the continuous appliance measurements of BLOND and FIRED in advance. This was achieved using the event detector introduced in Section 7.1.1. It is further noted that solely appliance switch-on events are used in this evaluation. These were obtained from the set of all events by selecting only events for which the average power within 3 s before the event is below an idle power. The idle power was calculated as 3 VA plus the smallest power spotted before any event of the particular

appliance. To further increase the comparability between all datasets, WHITED and PLAID events were also extracted using the same event detector, although these datasets already provide isolated appliance switch-on events. However, this ensured that the same event detection threshold of 3 VA was applied to all events of the evaluation set.

For each dataset, all events are shuffled and split into 80 % training and 20 % test samples (stratified). Using grid search, the model parameters of each classifier (see Table 7.1) are optimized on each training set (i.e., dataset) individually. In addition, a random stratified 5-fold CV is applied during grid search and the results are averaged across all folds to optimize for better generalization performance. The splits are performed stratified so that the class distributions remain constant across all CV splits. The model is then selected for each dataset and classifier based on the average CV results. The final model performance is ultimately reported for the test set after being trained on the complete training set.

Model selection with grid search and the final evaluation uses the confusion matrix notation in terms of TP, TN, FP, and FN to calculate accuracy (Equation 2.19), precision (Equation 2.20), recall (Equation 2.21), and F_1-score (Equation 2.22). Macro-averaging is used and the unweighted means of each metric is calculated. This means that all classes contribute equally to the mean of each metric ensuring that a class with more support in terms of the available number of samples (i.e., events) does not bias the results. To simplify the evaluation, two different appliances of the same type are treated as the same target class (e.g., the appliances *Monitor Dell U2713H* and *Monitor Fujitsu P171* of the WHITED dataset are regarded as the same class *monitor*). Classes with a support of less than five samples are removed from the evaluation to ensure that training will include at least four samples (for a 80/20 *train-test* split).

7.2 Standalone Feature Evaluation

In a first step, each feature is evaluated individually by training each classifier solely on a single feature. As HPO is performed for each classifier, each dataset, and each feature individually, a total of $4 \cdot 4 \cdot 27 = 432$ different grid search instances are evaluated. The final results are reported in Table 7.3 and represent the F_1-scores of the selected models applied to the test set.

The results show that some features alone (e.g., VIT, WFA, COT, or HED) already show decent classification capabilities (F_1-score > 0.8) while other features like PNR or PSS stand out with exceptionally poor F_1-scores. In the time domain, e.g., the VIT already reached an F_1-score of 0.99 and 0.95 on the laboratory datasets WHITED and PLAID, respectively. Those high scores could not be matched

for the FIRED and BLOND datasets which represent data closer to a real-world scenario. In the frequency domain, the HED achieves comparatively high scores of 0.97 on WHITED and PLAID while again not matching this performance on FIRED and BLOND (0.89/0.8). LAT, PNR, MAMI, MIR, PSS, and SPF show a very low average F_1-score ($\oslash < 0.2$). As found by Kahl et al. [117] among others, these features may be bad at distinguishing a larger set of different appliances but can be used to recognize specific appliances which exhibit certain electrical characteristics. Interestingly, those features (except MAMI) show consistent better results on BLOND and PLAID compared to FIRED and WHITED. Both BLOND and PLAID have a larger inner-class variability compared to FIRED and WHITED indicating that these features might still improve classification performance if more data are available for training. Unsurprisingly, features which show better performance have the drawback of a high dimensionality (e.g., 40 for VIT and 20 for WFA). If the focus is shifted towards the best performing scalar features (F_1-score > 0.4), classical electrical features such as P, S, R, Y, $cos\Phi$, and THD can be identified. It is argued that these features may be of choice for lightweight NILM algorithms deployed on resource constrained systems such as smart meters.

Table 7.3: Classification results of a single feature applied to each dataset (WHITED, PLAID, FIRED, and BLOND) using four classifiers (*k*NN, SVM, RF, and XGBoost (xgb)). HPO using grid search and 5-fold CV has been applied. The features with the highest F_1-scores for each dataset are highlighted bold in the time and spectral domain, respectively.

		WHITED				PLAID				FIRED				BLOND				
Feature	Dim.	knn	svm	rf	xgb	knn	svm	rf	xgb	knn	svm	rf	xgb	knn	svm	rf	xgb	⊘
time domain																		
Active Power (P)	1	0.49	0.45	0.48	0.51	0.58	0.53	0.56	0.57	0.65	0.62	0.63	0.66	0.5	0.49	0.49	0.49	0.54
Reactive Power (Q)	1	0.29	0.3	0.31	0.32	0.37	0.41	0.43	0.34	0.47	0.53	0.54	0.52	0.36	0.32	0.36	0.34	0.39
Apparent Power (S)	1	0.53	0.49	0.48	0.5	0.45	0.46	0.43	0.43	0.59	0.62	0.62	0.6	0.41	0.42	0.41	0.43	0.49
Resistance (R)	1	0.52	0.5	0.49	0.55	0.43	0.4	0.44	0.46	0.68	0.55	0.66	0.65	0.41	0.42	0.4	0.42	0.5
Admittance (Y)	1	0.51	0.5	0.49	0.55	0.43	0.43	0.44	0.46	0.68	0.63	0.66	0.65	0.41	0.41	0.39	0.42	0.5
Crest Factor (CF)	1	0.15	0.17	0.17	0.18	0.38	0.36	0.32	0.39	0.33	0.33	0.32	0.31	0.42	0.31	0.41	0.42	0.31
Form Factor (FF)	1	0.27	0.22	0.26	0.26	0.44	0.44	0.43	0.46	0.36	0.3	0.37	0.37	0.34	0.34	0.35	0.33	0.35
Log Attack Time (LAT)	1	0.05	0.05	0.05	0.05	0.12	0.16	0.15	0.15	0.1	0.09	0.09	0.09	0.19	0.22	0.23	0.19	0.12
Temporal Centroid (TC)	1	0.15	0.17	0.15	0.15	0.38	0.42	0.37	0.42	0.3	0.22	0.23	0.22	0.25	0.23	0.27	0.25	0.26
Positive-Negative Half Cycle Ratio (PNR)	1	0.04	0.03	0.09	0.05	0.19	0.18	0.16	0.19	0.19	0.19	0.2	0.19	0.18	0.15	0.18	0.12	0.14
Max-Min Ratio (MAMI)	1	0.06	0.06	0.06	0.07	0.28	0.26	0.18	0.25	0.28	0.3	0.29	0.32	0.13	0.11	0.12	0.11	0.18
Peak-Mean Ratio (PMR)	1	0.2	0.19	0.16	0.19	0.4	0.3	0.43	0.37	0.38	0.37	0.37	0.39	0.4	0.38	0.42	0.38	0.33
Max-Inrush Ratio (MIR)	1	0.07	0.04	0.07	0.06	0.16	0.17	0.16	0.14	0.12	0.12	0.11	0.1	0.13	0.13	0.14	0.12	0.12
Mean-Variance Ratio (MVR)	1	0.21	0.24	0.3	0.28	0.41	0.34	0.36	0.38	0.42	0.41	0.4	0.4	0.33	0.34	0.35	0.32	0.34
Waveform Distortion (WFD)	1	0.27	0.24	0.24	0.23	0.35	0.36	0.38	0.39	0.44	0.44	0.43	0.45	0.3	0.31	0.28	0.3	0.34
Period to Steady State (PSS)	1	0.01	0.03	0.03	0.03	0.11	0.12	0.12	0.12	0.09	0.12	0.12	0.12	0.11	0.12	0.12	0.12	0.09
Phase Angle ($cos\Phi$)	1	0.26	0.24	0.26	0.27	0.48	0.46	0.42	0.49	0.45	0.45	0.43	0.44	0.43	0.44	0.38	0.44	0.4
Inrush Current Ratio (ICR)	1	0.17	0.07	0.15	0.16	0.27	0.23	0.27	0.36	0.41	0.22	0.44	0.37	0.27	0.25	0.26	0.25	0.26
Waveform Approximation (WFA)	20	0.92	0.91	0.93	0.83	0.93	0.92	0.9	0.9	0.91	0.93	0.88	0.84	0.84	0.82	0.75	0.8	**0.88**
Current Over Time (COT)	25	0.8	0.84	0.93	0.86	0.81	0.72	0.86	0.87	0.88	0.93	**0.95**	0.94	0.8	0.81	0.83	0.83	0.85
V-I Trajectory (VIT)	40	0.93	**0.99**	0.95	0.89	0.91	0.93	**0.95**	0.88	0.7	0.82	0.77	0.72	0.82	**0.85**	0.71	0.78	0.85
spectral domain																		
Total Harmonic Distortion (THD)	1	0.37	0.39	0.34	0.37	0.49	0.5	0.48	0.51	0.42	0.4	0.43	0.41	0.38	0.34	0.39	0.38	0.41
Spectral Flatness (SPF)	1	0.06	0.07	0.09	0.1	0.2	0.19	0.21	0.22	0.17	0.17	0.15	0.18	0.23	0.21	0.19	0.21	0.17
Odd-Even Harmonics Ratio (OER)	1	0.09	0.09	0.12	0.09	0.26	0.28	0.26	0.3	0.28	0.25	0.26	0.25	0.29	0.25	0.3	0.21	0.22
Spectral Centroid (SC)	1	0.12	0.12	0.13	0.14	0.31	0.3	0.25	0.26	0.22	0.2	0.22	0.23	0.33	0.31	0.29	0.3	0.23
Tristimulus (TRI)	3	0.89	0.86	0.86	0.79	0.87	0.82	0.82	0.79	0.77	0.81	0.84	0.77	0.61	0.64	0.63	0.59	0.77
Harmonic Energy Distribution (HED)	18	0.97	0.93	**0.97**	0.83	**0.97**	0.88	0.94	0.93	0.85	0.85	**0.89**	0.88	0.7	**0.8**	0.77	0.77	**0.87**

7.3 Feature Selection

Some of the features already performed quite well in the standalone analysis. However, it can be assumed that the combination of multiple features leads to even better classification scores. While combining all 27 features may result in better classification performance, the number of dimensions should be held small to save computational resources and to prevent performance degradations which stem from larger feature spaces also known as *the curse of dimensionality*. Therefore, in a second analysis several feature combinations are evaluated based not only on their final classification score but also on their overall dimensionality. While the standalone feature VIT already reaches an F_1-score of up to 0.99 in the experiments, its large dimensionality may hamper a possible application. Furthermore, it might be possible that a combination of multiple features of smaller dimensionality even outperforms VIT. Consequently, a second analysis is conducted for which the combination of several features up to a maximum dimensionality N is examined.

Since an evaluation that considers all possible feature combinations is not feasible ($\sum_{k=0}^{27} \binom{27}{k}$), a simple sequential selection algorithm is used. The algorithm starts by adding the best performing scalar feature ($feat^x$) to a feature set ($F_0 = \{feat^x\}$ with dimensionality $N_0 = 1$). It then evaluates all combinations of F_i with another scalar feature $feat^j$. The best performing combined set ($F_{i+1} = F_i \cup \{feat^j\}$) is stored which results in a dimensionality of $N_{i+1} = N_i + 1$. It is then checked if any of the possible combinations of non-scalar features (F_{i+1}^{NS}) which result in the same dimensionality N_{i+1} outperforms F_{i+1}. If this is not the case, the algorithm continues with F_{i+1}, otherwise F_{i+1}^{NS} is used. This process is repeated until a maximum dimensionality N_{max} is reached. The performance of each tested feature set is stored.

The selection scheme is executed for all 27 features on all datasets with a kNN, SVM, and RF classifier. XGBoost was left out due to its extensive computational requirements and comparable low performance on the standalone feature evaluation (see Table 7.3). The results of this experiment, which are visualized in Figure 7.7, highlight that feature combinations with rather low dimensionality ($N < 10$) already lead to classification scores of over 98 % on WHITED and PLAID. The evaluation further highlights that the performance on recordings in laboratory setups (PLAID and WHITED) is generally better and more consistent compared to more representative real-world data (FIRED and BLOND). This is, however, expected due to the lower noise-level in such environments.

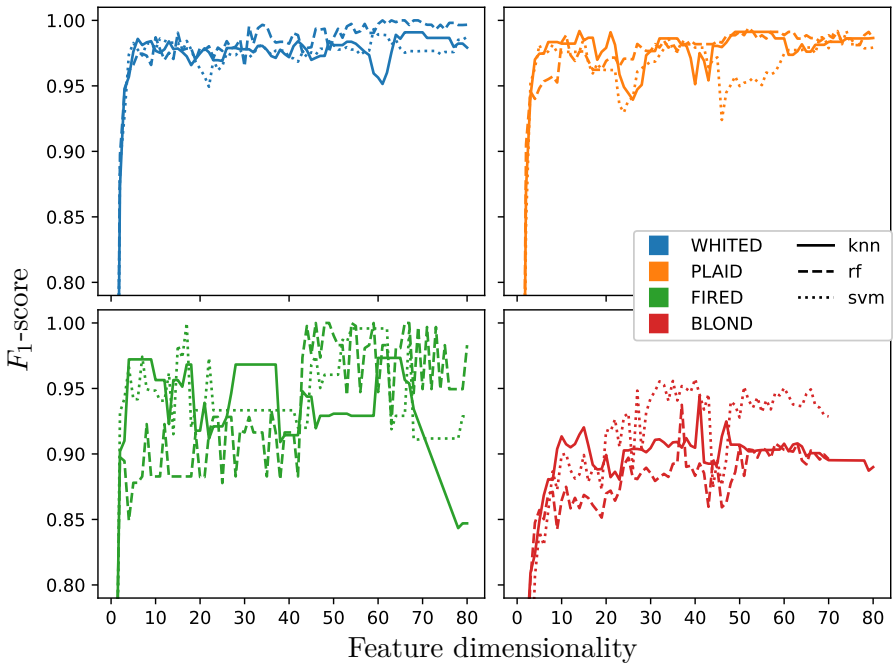

Figure 7.7: Results of the proposed feature selection strategy for all classifiers (line styles) and all datasets (line colors).

In this evaluation, all classifiers performed equally well. Only for the BLOND dataset, SVM classifiers outperform the others by quite a margin. Table 7.4 shows the specific feature sets that have been chosen by the selection scheme for different dimensionalities N.

As a tradeoff between dimensionality, performance, and computational effort, it is proposed to use features up to a dimensionality of 25. The feature set which has been proposed by the algorithm for $N = 25$ (see Table 7.4) depends on the used classifier. However, it always includes the features WFA and TRI. It is decided to supplement these features with P and $cos\,\Phi$ resulting in the proposed feature set $[P, cos\,\Phi, TRI, WFA]$. P has already been evaluated in Table 7.3 as being the best scalar feature with an average F_1-score of 0.54. $cos\,\Phi$ reoccurs in nearly all feature sets (see Table 7.4) and is added to accommodate the reactive component which may be introduced by an appliance. TRI further showed high classification results in Table 7.3 and represents the only frequency domain feature in the set. TRI is preferred over the actually better performing HED (see Table 7.3), as it requires only three dimensions instead of 18. From the corresponding formulas, it can be seen that TRI also represents a compressed form of the HED feature. While WFA (with a dimensionality of 20) did not outperform 20 scalar features, its

simple calculation (cf. Equation 7.17) and the overall best results obtained in the standalone feature analysis (see Table 7.3) justifies its inclusion in the set which is finally proposed.

Table 7.4: Used features for selected dimensionalities N of the proposed feature selection strategy. The F_1-scores for each dataset and classifier (Clf.) are shown in addition to the F_1-scores averaged over all datasets.

N	Clf.	Featureset	WHITED	PLAID	FIRED	BLOND	$\oslash F_1$
2	knn	P, Y	0.87	0.89	0.89	0.75	0.85
2	svm	P, Y	0.89	0.79	0.9	0.74	0.83
2	rf	P, Y	0.86	0.83	0.85	0.71	0.81
5	knn	$P, Y, THD, cos\Phi, OER$	0.98	0.97	0.94	0.82	0.93
5	rf	$P, Y, THD, cos\Phi, MVR$	0.96	0.96	0.93	0.77	0.91
5	svm	$P, Y, THD, cos\Phi, MVR$	0.96	0.98	1.0	0.81	0.93
22	knn	$P, Y, THD, cos\Phi, OER, FF, R, S,$ $PMR, WFD, TC, ICR, MVR, SC, Q, CF,$ $PNR, MAMI, SPF, LAT, PSS, MIR$	0.87	0.96	0.91	0.89	0.91
22	rf	$P, Y, THD, cos\Phi, MVR, WFD, SC,$ $LAT, OER, SPF, ICR, CF, S, PSS,$ $MIR, PNR, R, MAMI, Q, TC, PMR, FF$	0.98	0.97	0.92	0.87	0.93
22	svm	$P, Y, THD, cos\Phi, MVR, CF, R,$ $OER, FF, SC, Q, PMR, S, WFD, SPF,$ $ICR, TC, PNR, MAMI, LAT, PSS, MIR$	0.95	0.96	0.97	0.88	0.94
25	knn	$WFA, TRI, cos\Phi, Y$	0.99	1.0	0.92	0.9	0.95
25	rf	WFA, TRI, LAT, S	0.98	0.98	0.91	0.88	0.94
25	svm	WFA, TRI, S, THD	0.98	0.95	0.93	0.95	0.95
32	knn	$WFA, TRI, cos\Phi, Y, R, OER,$ MVR, THD, P, S, SC	0.99	1.0	0.93	0.89	0.95
32	svm	$WFA, TRI, S, THD, R, CF, Y,$ $cos\Phi, MVR, FF, OER$	0.98	0.97	0.93	0.96	0.96
32	rf	$WFA, TRI, LAT, S, CF, Y, MAMI,$ SC, SPF, MIR, OER	0.98	0.97	0.91	0.89	0.94

With these four features, the proposed set is of comparatively small dimensionality, computationally lightweight enough for resource constrained systems, and still delivers decent classification results. To emphasize this, the proposed set and the combination of all 27 features was evaluated on all classifiers and datasets. The results are shown in Table 7.5. A slight performance increase across all classifiers can even be identified if the proposed feature set is used instead of all features. This is due to the *curse of dimensionality*. With a dimension reduction from 128 to 25, the proposed set still outperforms the combination of all features, highlighting the effectiveness of the proposed feature set.

Table 7.5: Classification results for all 27 features and for the proposed feature combination $[P, cos\,\Phi, TRI, WFA]$. The best results are highlighted in bold.

	WHITED				PLAID				FIRED				BLOND				
Clf	Pr	Re	Ac	F_1	Pr	Re	Ac	F_1	Pr	Re	Ac	F_1	Pr	Re	Ac	F_1	$\oslash F_1$
using all 27 feature; overall vector dimension: 128																	
knn	0.97	0.97	0.96	0.97	0.98	0.97	0.97	0.97	0.89	0.87	0.99	0.86	0.94	0.94	0.99	**0.94**	0.93
svm	0.99	0.99	0.99	0.99	0.99	0.98	0.98	0.98	0.91	0.94	1.0	0.92	0.96	0.88	0.99	0.91	0.95
rf	1.0	1.0	0.99	**1.0**	0.99	0.98	0.98	**0.98**	1.0	0.98	1.0	**0.98**	0.97	0.88	0.99	0.91	**0.97**
xgb	0.97	0.96	0.97	0.96	0.97	0.96	0.97	0.97	0.9	0.87	0.99	0.87	0.95	0.86	0.99	0.89	0.92
using the feature set $[P, cos\,\Phi, TRI, WFA]$; overall vector dimension: 25																	
knn	0.97	0.97	0.96	0.97	0.98	0.98	0.98	0.98	0.91	0.92	0.99	0.91	0.95	0.89	0.99	0.91	0.94
svm	0.98	0.97	0.97	0.97	0.96	0.96	0.95	0.96	0.92	0.95	1.0	0.93	0.94	0.96	0.99	0.95	0.95
rf	0.99	0.99	0.98	**0.99**	1.0	0.99	0.99	**0.99**	1.0	0.98	1.0	**0.98**	0.96	0.95	0.99	**0.95**	**0.98**
xgb	0.94	0.91	0.93	0.91	0.97	0.96	0.97	0.97	1.0	0.97	1.0	0.98	0.95	0.86	0.98	0.89	0.94

The average F_1-scores of the proposed set over all datasets exceed 0.94 independent of the used classifier. The RF classifier performs best with an average F_1-score of 0.98. It is, however, noted that a computationally fairly simple kNN classifier with $k = 1$ already achieves a rather high F_1-score of 0.97 on WHITED and 0.98 on PLAID. kNN is a so called *lazy learning* algorithm that requires no internal parameter tuning except for the choice of the number of neighbors (k) to consider. During training, the complete training set is stored. During inference, a new sample is assigned to the most common class within its k-nearest neighbors. To reduce the required memory of a kNN classifier, which linearly increases with the number of training samples, the Condensed Nearest Neighbor Rule [169] can be applied. Because of its simple training and the ability to reduce the required memory, it is argued that kNN should be the classifier of choice if deployed (including training) on systems with small computational resources such as typical smart meters. It might also be possible to integrate a kNN classifier into the SmartMeter which has been proposed in Section 4.2. However, for systems with sufficient computational power, SVM and RF should be the classifiers of choice.

XGBoost has shown enormous potential by leading many machine learning competitions during the recent years [170]. Even though it exhibits the worst performance across all classifiers in the analysis at hand, it is argued that XGBoost might still outperform RF and SVM for other hyperparameter choices as the ones tested during these evaluations (see the used grid search parameters in Table 7.1). However, due to its large hyperparameter space and, therefore, extensive training time, RF and SVM were selected in favor, representing a tradeoff between the required training time and possible gain in classification performance.

Figure 7.8 shows the confusion matrix of the RF classifier using the proposed feature set on the PLAID dataset (the corresponding performance metrics are shown in Table 7.5. Despite the overall F_1-score of 0.98, only some appliances with rotary motors (fan, heater, and air conditioner) are confused with one another.

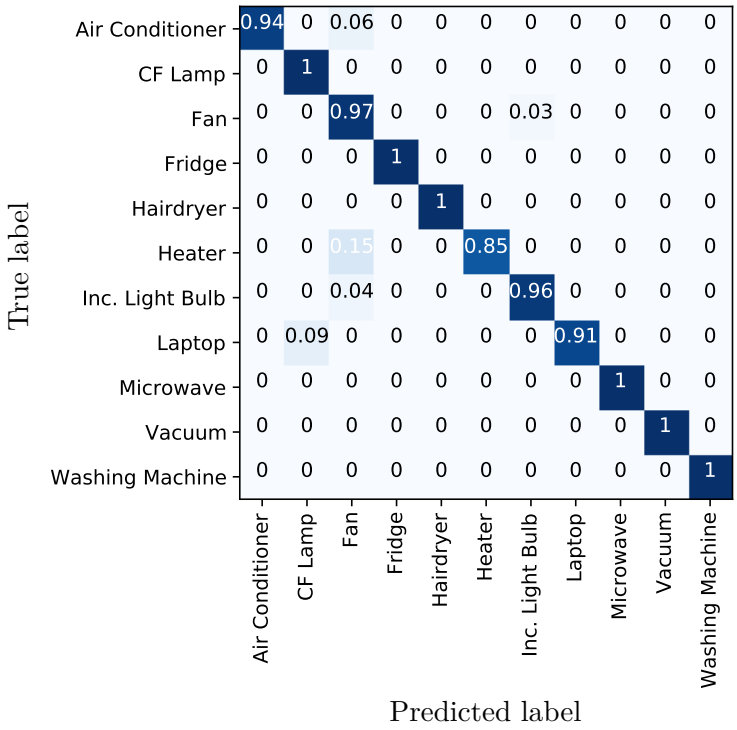

Figure 7.8: Confusion matrix of a RF classifier with the feature set [P, $cos\,\Phi, TRI, WFA$] applied to the PLAID dataset.

Due to the outstanding performance of the RF classifier with the feature set [P, $cos\,\Phi, TRI, WFA$], it is proposed to use their combination as a benchmarking algorithm when comparing novel appliance classification algorithms, similar to the low-frequency disaggregation algorithms that have been implemented as benchmarks in NILMTK [69].

The experiments presented in this chapter solely used start-up events extracted from individual appliance data (such as recorded by a PowerMeter). However, to apply the basic concepts of the event-based NILM pipeline (cf. Figure 2.2) to aggregate data, i.e., the data recorded by smart meters, additional adjustments to the data and the training procedure are required. These are described in more detail at the beginning of Chapter 8.

8 Improving Supervised Non-Intrusive Load Monitoring

The general pipeline of event-based NILM systems is shown in Figure 2.2. Data acquisition has been approached in Chapter 4, event detection has been improved in Chapter 5, and well-established features and classifiers for event classification have been evaluated in Chapter 7. This chapter introduces several cleaning methods that can be used to exempt an appliance event from the aggregated consumption data. Afterwards the concept of hybrid training, in the context of NILM systems, is explained and evaluated. Finally, a novel supervised NILM method is introduced which is based on the combination of aggregated and individual appliance data. The techniques explained in this chapter have been published in [PA21].

- A comparison of different cleaning methods tailored for the task of extracting individual appliance events from the aggregated consumption data (see Section 8.1).
- A comparison of naive, classic, and hybrid supervised training (see Section 8.2).
- The introduction of a novel approach for supervised NILM using additional plug-level meters (see Section 8.3).

8.1 Event Cleaning

In contrast to Chapter 7 where isolated high-frequency measurements of appliance events were used to calculate characteristic features, classical high-frequency event-based NILM systems only have access to aggregated (voltage and current) measurements. Appliance events can still be detected and compressed into an ROI using aggregated measurements. These measurements will, however, contain contributions of all currently active appliances, not just the contribution of the appliance causing the event. While it can be assumed that the voltage level at each appliance is virtually identical to the level measured at the aggregated point, the current measured at this point is the sum of the currents drawn by each appliance

(cf. Equation 2.2). If the features explained in Section 7.1.2 are extracted from aggregated appliance events, the contributions of other appliances may distort the characteristics of the appliance causing the event. A simplified example: If the active power consumption was 1000 W before and is 1100 W after the event, the contribution of the appliance causing the event is just +100 W. Therefore, special cleaning techniques are required so that the features (active power in the example) only characterize the appliance causing the event (+100 W in the example) and not all currently active appliances (1100 W in the example). Cleaning can either be applied to the aggregated current and voltage measurements or to the calculated features, resulting in several possibilities which are evaluated in the following.

Subtracting the characteristic waveform:
For this technique a prototype mains cycle I_{WF}^{proto} is calculated by averaging all mains cycles from the pre-event data. This prototype waveform is subtracted from all mains cycles following the event after cross correlating the individual mains cycles. The cleaned mains cycles are calculated as $I_{WF}^{cleaned}[j] = I_{WF}[j] - I_{WF}^{proto}$.

Subtraction in the frequency domain:
The pre- and post-event data are first transformed to the frequency domain using a Fourier transformation (cf. Equation 7.21). Afterwards the magnitudes of the frequency bins are subtracted as $X[k] = X^{post}[k] - X^{pre}[k]$. Finally, the signal is transformed back to the time domain using an inverse Fourier transformation.

An example of both cleaning methods is shown in Figure 8.1. The top plot shows the ROI of a fridge event of the FIRED dataset taken from the aggregated data. The corresponding isolated recording is shown in the bottom plot. The orange signals represent the two cleaning methods and have been generated solely using the (blue) aggregated signal. As the goal of the cleaning method is to emphasize the contribution of the individual appliance causing the event, a result closer to the isolated recording (green) indicates a better cleaning. Both methods effectively remove the steady-state current prior to the event from the signal. To rate the similarity to the isolated measurement, the RMSE (see Equation 2.18), MAE (see Equation 2.17), and the Pearson correlation coefficient ($C_{pearson}$) is used. $C_{pearson}$ rates the similarity of two sampled signals x and y w.r.t. to their correlation, and is defined as

$$C_{pearson} = \frac{\sum_{i=1}^{N}(x[i] - \mu_x)(y[i] - \mu_y)}{\sqrt{\sum_{i=1}^{N}(x[i] - \mu_x)^2}\sqrt{\sum_{i=1}^{N}(y[i] - \mu_y)^2}}, \tag{8.1}$$

where μ_x and μ_y are the means of the signals x and y which are of length N. $C_{pearson}$ is independent of an applied linear mapping to the signal. Hence, if x

and y are perfectly mapped as $y = n \cdot x + m$ (with scaling factor n and offset m), the corresponding correlation coefficient is 1. In the context of this evaluation this might look counterintuitive at first, as the contributions of other appliances are of additive nature. However, waveform distortions, such as those caused by phase shifts and non-resistive behaviors of other appliances, are still considered by the correlation coefficient. When comparing the similarity to the individual measurement, a cleaned signal x is compared to the individual measurement y. $C_{pearson}$ has been calculated for each signal in Figure 8.1 (see the legend of the plots). In case of identity ($x = y$), $C_{pearson}$ evaluates to 1 representing optimal correlation (see $C_{pearson}$ for green signal). The better the similarity of the signals, the closer $C_{pearson}$ is to 1. Hence, the cleaning methods perform equally well (see $C_{pearson}$ for both orange signals in Figure 8.1), with a slightly better correlation coefficient for the frequency domain method.

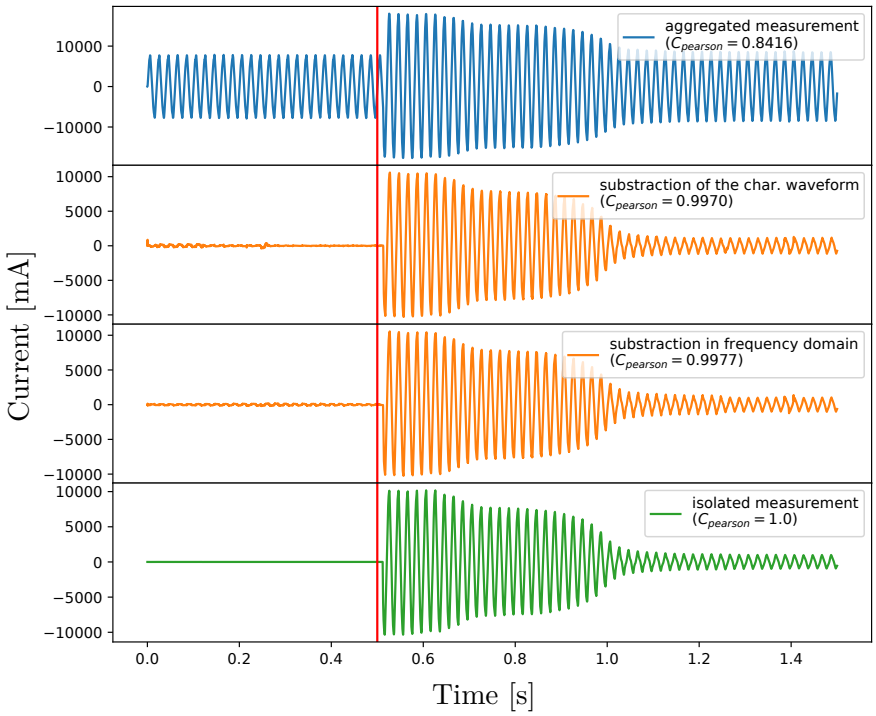

Figure 8.1: (Blue) fridge event found in the aggregated current signal of FIRED. (Orange) cleaned events using the current signal of the first 0.5 s. (Green) recording of the same event at appliance-level.

To further test the cleaning methods, they were applied on several other appliance events from the FIRED dataset. The correlation coefficient, the RMSE, and the MAE have been calculated alongside. The averaged results are shown in Table 8.1. In total, 192 events from 23 different appliances have been used.

Table 8.1: Comparison of the two proposed cleaning methods for aggregated data. *None* refers to no cleaning and acts as a baseline.

Cleaning method	Pearson	MAE	RMSE
None	0.84	666.01	2360516.75
Time domain	0.86	291.39	458307.01
Frequency domain	0.81	334.97	458036.25

While no cleaning seems to have an even better Pearson coefficient than frequency domain cleaning (potentially due to its invariance against scaling), the corresponding average MAE and RMSE values are noticeable smaller if cleaning is applied (smaller is better for both metrics). Time domain cleaning performs best according to the Pearson and MAE metric. Still, the results highly depend on the signal prior to the event. If the signal is disturbed by other events or contains other non-stationary *noise* due to the contributions of other appliances, the cleaning efficiency will decrease. As the electricity consumption of an appliance is generally time-variant and non-stationary (as it is user and time dependent), no cleaning method will be able to fully reconstruct the isolated measurements.

Instead of cleaning the original measurements to subsequently calculate the features from the cleaned measurements, the corresponding delta representations can be calculated directly in the feature domain.

Delta Form:
As most steady-state features meet the feature additive criterion, the delta form of each feature can be used as $feat^{\Delta} = feat^{post} - feat^{pre}$ to isolate the contribution of the appliance causing the event. Other features require corresponding adaption (such as a subtraction in the frequency domain for the THD feature) before a delta representation can be calculated. Similarly, most transient features require adjustments, such as the COT, where the corresponding steady-state current from before the event needs to be subtracted.

To finally decide on the best event cleaning strategy, a small evaluation compares the time domain, frequency domain, and delta form cleaning methods. Labeled aggregated events from the FIRED dataset are first cleaned with a particular method and are then used to calculate all 27 features presented in Section 7.1.2. While the proposed feature subset in Section 7.3 achieved very good results, this evaluation aims to analyze the effect of the cleaning methods on the performance of all features. Hence, all features are used to train and test the four classifiers introduced in Section 7.1.3. This results in an overall feature vector of size 128.

The used training and evaluation procedure is similar to the one described in Section 7.1.4 utilizing grid search with 5-fold CV to optimize the hyperparameters listed in Table 7.1 as well as the macro averaged precision, recall, accuracy, and F_1-score metrics. The results are presented in Table 8.2 and indicate that all cleaning methods yield a noticeable gain in classification performance compared to no cleaning. Time domain cleaning shows the best results. The delta form also performs quite well and frequency domain cleaning performs slightly worse but still almost 5 % better than no cleaning. Hence, whenever aggregated data are used in the following, the time domain cleaning method will be applied to the data before any feature is extracted from it.

Table 8.2: Comparison of the classification results for a *k*NN, an SVM, and a RF classifier after applying one of the proposed cleaning methods. *None* refers to no cleaning.

Cleaning method	knn				svm				rf			
	Pr	Re	Ac	F_1	Pr	Re	Ac	F_1	Pr	Re	Ac	F_1
None	0.71	0.7	0.95	0.71	0.81	0.79	0.97	0.79	0.87	0.83	0.97	0.84
Time domain	0.84	0.82	0.97	**0.82**	0.97	0.93	0.99	**0.94**	0.99	0.98	0.99	**0.98**
Freq. domain	0.83	0.82	0.96	0.82	0.95	0.9	0.98	0.91	0.9	0.89	0.98	0.89
Delta form	0.77	0.79	0.97	0.77	0.92	0.88	0.98	0.88	0.96	0.94	0.99	0.94

Table 8.2 further shows that RF achieved the best classification score in this evaluation, outperforming SVMs by around 4 %. This confirms the results shown in Table 7.5 where the performance of all features is compared to the proposed feature set (see Section 7.3). RF can inherently handle multiple feature dimensions, as a dedicated hyperparameter ($max_{features}$) reduces the number of features per split [171]. The standard value for $max_{features} = \sqrt{n_{features}}$ already reduces the feature dimensionality of each split from 128 to 11.

8.2 Hybrid Training

This section aims to answer whether additional training on high-frequency individual appliance data improves the classification performance compared to training solely on labeled aggregated data. This method is referred to henceforth as *hybrid training*. Both, the FIRED and the BLOND dataset offer simultaneous high-frequency recordings of the aggregated mains signal and of many individual appliances. This allows to extract appliance events (i.e., the ROIs of the events) from individual appliance data and aggregated data. As the corresponding ROIs

include the same appliance specific characteristics, a combined use is evaluated for the task of model training.

The following evaluation is performed for three scenarios. The classifiers are trained (*S1*) solely on individual appliance recordings, (*S2*) solely on aggregated recordings, and (*S3*) on aggregated and individual appliance data. While training may include individual appliance recordings (for scenario *S1* and *S3*), testing is performed on cleaned aggregated data only as this represents a NILM system's target input data after training. Whenever aggregated data are used, these data are cleaned using the time domain cleaning method proposed in Section 8.1.

Scenario *S1* is motivated by the fact that *clean* and appliance-specific recordings are used for training which are not disturbed by noise from other appliances as it might be the case for aggregated recordings. Scenario *S2* trains the system on target data only and might learn the specific noise distribution of the data and measurement system. This scenario, therewith, represents standard supervised NILM setups. Scenario *S3* is motivated by the combination of *S1* and *S2*. If the training uses both *clean* and target data, a classifier may better generalize for non-stationary noise in the data. Furthermore, twice the amount of samples are available for training as each event is recorded at individual and aggregated level. Therewith, this scenario can also be seen as a kind of data augmentation technique. To see the impact of doubling the number of training samples in *S3*, an additional evaluation (*S3.5*) was performed using only a subset of the training data which has the same size as the training sets of *S1* and *S2*.

8.2.1 Data Preparation and Evaluation Setup

To extract events from the aggregated data from both, the FIRED and the BLOND dataset, the timestamps of the individual appliance events (which have already been used in Section 7.2 and Section 7.3) are employed. The exact event position in the aggregated recordings is determined by a fine search of a corresponding power change in a 3.5 seconds window applied to the event timestamps taken from the individual recordings. This allows for slight time shifts between the different DAQs (i.e., aggregated meter and plug meters) which have been observed to be less than 1 s in both datasets. As the aggregated signal measures the composite load of all appliances (on the same grid line), the aggregated signal of an appliance event can potentially be disturbed by additional events of other appliances. However, according to the SCP, the probability that more than one event happens simultaneously is very low. In order to confirm this, it was checked for simultaneous events within 3 s, as the ROI for an event is of length 1.5 s. It was observed, that 11 % of the start-up events in the considered time period of the FIRED dataset happened simultaneously, i.e., within 3 s of at least one other start-up event. Many of

these events were generated by a PC which frequently transitioned between an idle and active state. For the BLOND dataset, an even higher number of the start-up events (47 %) happened simultaneously with other events. This is mainly due to the comparably small event threshold of 3 VA (compared to larger thresholds used in other works, e.g., 25 W in [108]) and the specific environment of the BLOND dataset, where numerous variable power consumers and multi-monitor setups are used. For the following evaluation, all simultaneous events were removed from both datasets. This ensured that the extracted events are as *clean* as possible ensuring that the results mainly highlight the effect of the hybrid setup.

The same classifiers (kNN, SVM, RF, and XGBoost) as used in the standalone and combined feature analysis (see Section 7.2 and Section 7.3) are also utilized for this evaluation. Each classifier is trained for each scenario on each dataset. The proposed feature set from Section 7.3 is employed for all experiments. Since these features proved to work well with individual appliance data, they should also perform quite well with aggregated data, as ideally, aggregated data shows the same characteristics especially after cleaning (see Section 8.1).

8.2.2 Results

The results are summarized in Table 8.3. Overall, and independent of the specific scenario, the results highlight that a lower inner class diversity is beneficial for appliance classification, as the results for FIRED are significantly better than for BLOND. Scenario *S1*, in which only individual appliance data are used for training, shows quite poor results. While the kNN classifier still achieved an F_1-score of 0.58 for the FIRED dataset, no classifier was able to match such a score for the BLOND dataset. If the classifiers are trained on target data (*S2*), the results increase to e.g., 0.93 for FIRED and 0.82 for BLOND using a RF classifier. Finally, if individual and aggregated data are used for training, all classifiers tend to perform slightly better on average. With an average 7 % better F_1-score (compared to the next best classifier in *S3*), the SVM classifier outperforms all other classifiers and achieves the overall best average F_1-score of 0.82. It is noted that the training set for *S3* is doubled (compared to *S1* and *S2*), as all events are taken from individual and aggregated data. Therefore, in scenario *S3.5*, the training set was reduced to the same size as for *S1* and *S2* while still containing individual and aggregated data. The results show that the high scores of *S3* cannot be matched in this case, which indicates that the increase in training samples is the main benefit of the hybrid approach.

Summarized, training solely on individual recording (*S1*) delivers comparatively poor results (F_1-score < 0.6), training on aggregated data (*S2*) delivers decent

results ($0.7 < F_1$-score < 0.8), while training on both, individual and aggregated data, shows slightly better results ($0.73 < F_1$-score < 0.82). The better results of the latter have, however, been identified to be related to larger training sets.

Table 8.3: Average classification results for three different training sets using time domain cleaning, the proposed combination [$P, cos\,\Phi, TRI, WFA$], and four classifiers (kNN, SVM, RF, and XGBoost). *S1*: Training using appliance-level data only, *S2*: Training using aggregated-level data only. *S3*: Training on appliance-level and aggregated-level data. *S3.5*: Training on a subset of *S3* with the same size as *S2*.

		FIRED				BLOND				
Sc.	Clf.	F_1	Pr	Re	Ac	F_1	Pr	Re	Ac	⊘
S1	knn	0.58	0.66	0.62	0.94	0.37	0.46	0.49	0.49	0.47
	svm	0.53	0.56	0.59	0.93	0.22	0.42	0.23	0.26	0.38
	rf	0.57	0.64	0.59	0.91	0.25	0.29	0.29	0.36	0.41
	xgb	0.47	0.57	0.48	0.91	0.32	0.53	0.41	0.52	0.4
S2	knn	0.86	0.88	0.87	0.98	0.59	0.58	0.61	0.9	0.73
	svm	0.89	0.91	0.9	0.99	0.67	0.77	0.65	0.91	0.78
	rf	0.93	0.94	0.93	0.99	0.61	0.82	0.58	0.91	0.77
	xgb	0.79	0.8	0.8	0.99	0.56	0.64	0.55	0.91	0.67
S3	knn	0.87	0.89	0.88	0.99	0.63	0.65	0.62	0.91	0.75
	svm	**0.95**	0.96	0.96	0.99	**0.68**	0.79	0.64	0.92	**0.82**
	rf	0.94	0.94	0.93	1.0	0.53	0.54	0.52	0.91	0.73
	xgb	0.88	0.89	0.89	0.99	0.58	0.67	0.56	0.91	0.73
S3.5	knn	0.78	0.79	0.81	0.98	0.61	0.62	0.62	0.9	0.7
	svm	0.82	0.82	0.86	0.98	0.63	0.66	0.61	0.89	0.73
	rf	0.84	0.83	0.87	0.99	0.54	0.57	0.52	0.89	0.69
	xgb	0.84	0.84	0.86	0.99	0.55	0.61	0.53	0.91	0.69

8.3 Minimal-Intrusive Load Monitoring

Many supervised NILM systems require a preceding and intrusive training phase, during which all appliances must have been traversed through all of their possible states several times. For instance, one can think of a washing machine that has to be started several times for all of its possible washing programs. At best, this

should also be repeated for all other possible combinations with the states of all other appliances. This takes the term *non-intrusive* in NILM ad absurdum.

By combining the previously discussed improvements (cleaning and hybrid training) with a user-centric training phase supported by plug-level meters, a concept called Minimal-Intrusive Load Monitoring (MILM) is introduced. Based on the results from Section 8.2 it can be justified that training on individual and aggregated data maximizes the performance of supervised event-based NILM systems. Furthermore, no dedicated data labeling (i.e., ground truth generation process) is required if appliances are monitored individually. When an appliance event is recognized in the plug-level data, the corresponding event in the aggregated data can be automatically labeled without the need for any user interaction. Therefore, additional intrusive plug meters are used for (1) the collection of training samples not disturbed by noise, (2) the collection of ground truth labels for the aggregated data, and as (3) a straightforward way to guide the training process for the resident. This concept is called Minimal-Intrusive Load Monitoring as it combines the advantages of NILM and ILM while simultaneously resolving many disadvantages of conventional supervised NILM methods. It is considered to be *minimal* intrusive as it still requires some intrusive sensors during training and a minimal number of intrusive sensors after training. An overview of the MILM concept is shown in Figure 8.2 and the approach is explained in the following.

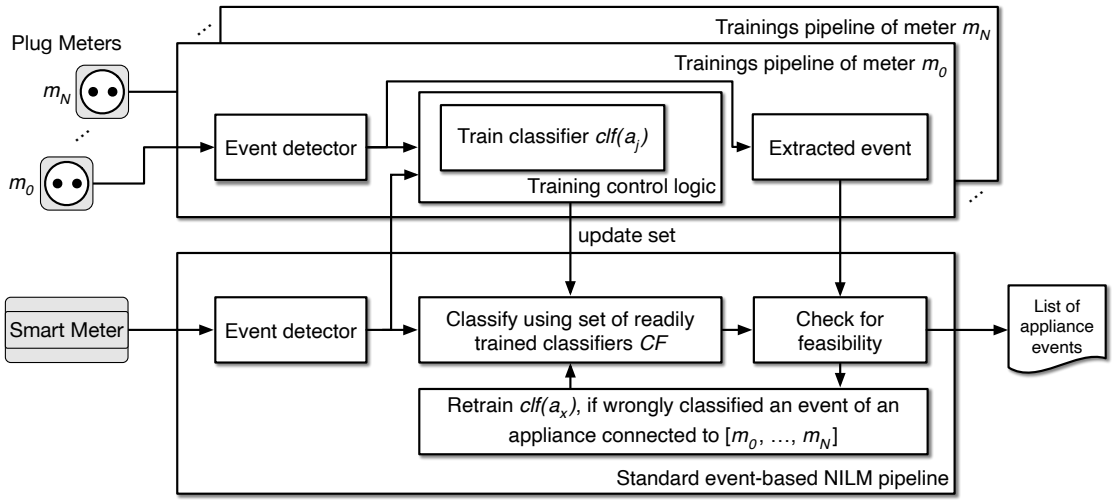

Figure 8.2: Concept of Minimal-Intrusive Load Monitoring with a set of N plug-level meters m_i which are connected to an appliances of interest a_j in a pipelining strategy, guided by a user-centric training phase.

Deploying the system in a new home requires the installation of a smart meter

capable of recording aggregated data at high sampling rates ($\geq 2\,\text{kHz}$) into the home's fuse box. Furthermore, several plug-level meters ($m_0, \ldots, m_N$) are handed to the resident, which can also record at a high sampling rate ($\geq 2\,\text{kHz}$). The control flow for each of these plug meters m_i is shown in Figure 8.3. To simplify the explanation of the concept, it is exemplarily considered that only a single plug meter is used in the following. The resident decides which appliance (a_j with $j \in [0, \ldots, M]$) is of most interest and installs the plug meter to it. The system then

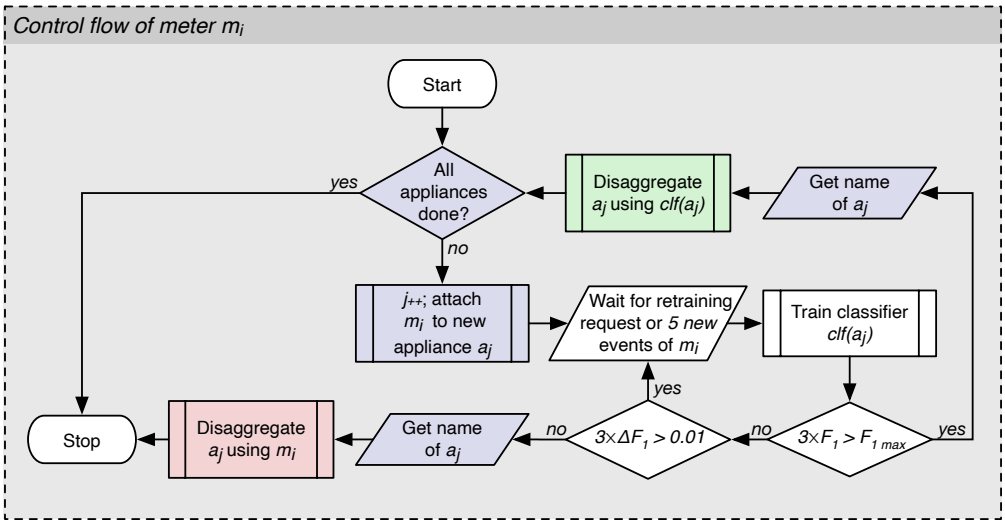

Figure 8.3: Flow of the training control logic of a single plug meter m_i. Green and red boxes indicate success and failure, respectively, and blue boxes require user attention.

collects an event set (ES_j) for appliance a_j using an event detection algorithm (see e.g., the detector introduced in Section 7.1.1). This set consists of the events of a_j taken from the plug-level and aggregated-level data. These are labeled as POS (=*positive*). Furthermore, all other events found in the aggregated-level data are added to ES_j and are labeled as NEG (=*negative*). If sufficient POS and NEG events are sampled (i.e., at least five samples of the positive and 20 samples of the negative class), ES_j is converted into an evaluation set ($EVAL_j$) for a binary classifier ($clf(a_j)$). ES_j is randomly subsampled into $EVAL_j$ to enforce a fixed class distribution of 80 % negative and 20 % positive events. There are naturally (but not necessarily) more negative than positive class instances with a beforehand unknown ratio, leading to an imbalanced classification problem. By restricting the class imbalance ratio to 1:4, resulting in a *slight imbalance* [66], it can still be handled by standard classification techniques and further includes sufficient variety of the negative class. $EVAL_j$ is further split into 20 % test samples ($TEST_j$) and 80 % training samples ($TRAIN_j$). All plug-level data are

removed from $TEST_j$ to ensure that the classifier is tested on unseen appliance events solely from the aggregated-level data. An SVM classifier (which showed the best results for *S3* in Section 8.2) is trained on $TRAIN_j$ using 5–fold CV. For every five new positive samples added to ES_j, or if a retraining is specifically scheduled (see below), training is repeated. Training stops if the classifier has reached an F_1-score of a certain threshold $F_{1\,max}$ for the last three retraining runs (*success*) or if the last three retraining runs have not improved the performance by at least 1 % (*failure*), as shown in the flow diagram depicted in Figure 8.3.

On *success*, the user is notified that training for appliance a_j has finished. To obtain a unique name for the appliance, the system might ask the user. As user interaction should be held minimal, this step might be further enhanced by e.g., providing a list of suggested appliance types. Those suggestions could be generated by applying the individual appliance data on general appliance models pre-trained on different homes. After a name is specified, plug meter m_i is connected to the next appliance of interest (a_{j+1}) for which the process is repeated.

On *failure*, events of appliance a_j cannot be adequately recognized by the system using aggregated-level data alone. The user is notified and prompted to leave the plug meter m_i installed to continue to meter the events of appliance a_j in an intrusive fashion. All events in the aggregated signal triggered by an appliance which is marked with *failure* are not included in the training set for other appliances and solely the plug-level meter is used to determine events of the connected appliance.

Each successfully trained classifier $clf(a_j)$ is added to a gradually increasing set of classifiers (CF) and the final training set $TRAIN_j$ is stored. CF is used in a *one-vs-rest* strategy to achieve multi-class classification for all appliances with a corresponding classifier in CF. For a given event, the classifier with the highest probability of the positive class determines the finally assigned class. If no classifier estimates a probability of at least 50 % for the positive class, the event is regarded as *unknown*. The events labeled by each individual meters m_i are further utilized to test all classifiers (depicted in Figure 8.2 as *check for feasibility*):

- If a currently trained classifier $clf(a_j)$ mistakenly classifies an aggregated event as an event of a_j (*false positive*), the aggregated event is added to an *explicit negative*-class and a retraining of $clf(a_j)$ is enforced. Explicit negative events are added to $EVAL_j$ independent of the positive-negative class distribution.
- If a currently trained classifier $clf(a_j)$ misses an aggregated event of a_j (*false negative*), it is retrained immediately.
- If a classifier $clf(a_x)$ in CF mistakenly classifies an aggregated event known to be an event of a currently metered appliance $a_j \neq a_x$ (*false positive*), it is added as an *explicit negative*-class to $TRAIN_x$ (the stored training set of $clf(a_x)$) and a retraining of $clf(a_x)$ is scheduled.

All classified or labeled events are added to a final event list which encompasses the timestamp and appliance corresponding to the events. As this also includes the events of all individually metered appliance, it allows for monitoring appliance events in case of *failure* and even during the training phase. Therewith, the proposed MILM system can determine information (i.e., timestamp and corresponding appliance) of the events generated by all appliances of interest. This is achieved by using the combination of a NILM system with a small number of (intrusive) plug-level meters to determine a final, minimal required number of plug meters for a given detection performance $F_{1\,max}$.

8.3.1 Simulation on FIRED

The proposed system was simulated on the third supply leg (L_3) on 71 days of data from the FIRED dataset (June 20 to August 30, 2020). The simulation solely used L_3 as the most individually metered appliances which actively trigger events are connected to it. In total, 25 of 31 appliances exhibit events in the FIRED data from which 19 appliances are connected to L_3. The implemented test-bed allows to simulate different $F_{1\,max}$, different numbers of plug meters (N), and different connection sequences (CS). A simulation of all possible combinations of CS and $F_{1\,max}$ is infeasible, since this would require $R = n \cdot \sum_{i=0}^{N} \binom{M}{i}$ simulation runs for the total number of appliances M and the n different $F_{1\,max}$ (e.g., $R = 2097152$ for $n = 4$ and $M = 19$). Instead, the order in which plug meters are attached to appliances is selected based on (1) the average appliance's power consumption during the simulation time period (CS_{power}) or (2) the number of events exhibited by each appliance (CS_{events}). The simulations are evaluated for a fixed count of five plug meter in terms of the overall performance (TP, FP, FN, TN) and the total number of appliances learned during the simulation time period. An appliance contributes to the overall performance as soon as its training has finished (*success*). Table 8.4 shows the simulation results for four different $F_{1\,max}$ thresholds, $N = 5$ and an SVM classifier.

It can be seen that a maximum of ten appliances can be learned within the given time period. This seems to be a comparably small portion of all 19 appliances but most of the remaining appliances simply do not trigger enough events during the simulation time period and could, therefore, not be learned successfully (see support column in Figure 8.5). The number of events of a particular appliance is larger if the appliance is either used more often (e.g., coffee machine compared to waffle maker), it triggers multiple events when turned on (e.g., the heating cycles of a dishwasher), or it triggers events without explicitly being turned on (e.g., the cooling cycles of a fridge). For this reason, it is generally more beneficial in terms of the system's learning rate to attach the appliances in an order

Table 8.4: Simulation of the proposed Minimal-Intrusive Load Monitoring system using an SVM classifier on the FIRED dataset. *f* and *s* represent the number of appliances for which learning *failed* and *succeeded*, respectively.

	CS_{events}						CS_{power}					
$F_{1\,max}$	f	s	Pr	Re	Ac	F_1	f	s	Pr	Re	Ac	F_1
0.80	0	9	0.89	0.93	0.97	0.91	0	**5**	0.87	0.93	0.97	0.91
0.85	0	9	0.89	0.93	0.97	0.91	0	4	0.89	0.89	0.97	0.9
0.90	0	**10**	0.92	0.92	0.97	0.92	1	3	0.94	0.97	0.97	0.97
0.95	4	2	0.99	0.99	0.99	**0.99**	2	2	0.99	1.0	0.98	**0.99**

determined by their exhibited number of events (CS_{events}) rather then their power consumption (CS_{power}). This is confirmed by the results in Table 8.4 which show that more appliances can be successfully learned over time ($s = success$) if the order is determined by the number of appliance events. Figure 8.4 also highlights this by showing the number of appliances learned over time for CS_{power} (dashed) and CS_{events} (non-dashed).

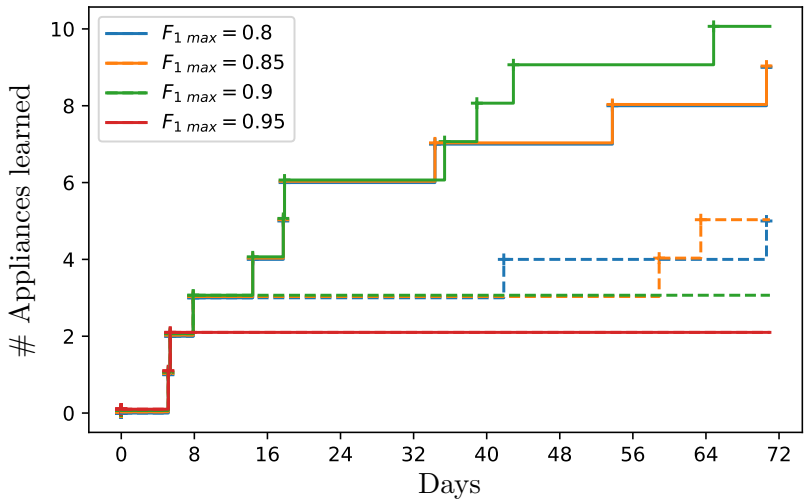

Figure 8.4: Number of learned appliances over time for different $F_{1\,max}$ and different connection sequences. Dashed lines represent CS_{power}, filled lines represent CS_{events}.

The dashed and non-dashed curves show similar behaviors in the first ten days of the simulation, as the same three appliances (*fridge*, *esspresso machine*, and

oven) are initially connected to the meters for CS_{events} and CS_{power}. After that, the number of appliances stays almost constant for CS_{power}, because the newly connected appliance (such as the *kitchen spot light*) do not trigger enough events to be learned, at least for the next 30 days. During this time period significantly more appliances are learned for CS_{events}, as the connected appliances trigger more events. It can be further seen that lowering $F_{1\,max}$ does not necessarily lead to an increase in the total number of appliances learned. It is noted that neither the number of events nor the power consumption of the appliances are known for real deployments in advance. Therewith, the order should ultimately be determined by the interest of the user. Lowering $F_{1\,max}$ generally leads to a faster learning rate and to an overall worse F_1-score. Therefore, a trade-off has to be made, which may also depend on user preference. If $F_{1\,max}$ is increased to 0.95, the highest overall score is achieved but only two appliances could be learned successfully. A reasonable tradeoff to balance classification performance and number of appliances learned is achieved for CS_{events} and $F_{1\,max} = 0.9$. While the overall F_1-score is lower compared to the simulation with $F_{1\,max} = 0.95$, the number of learned appliances is the highest of all.

Figure 8.5 shows the confusion matrix for the simulation run with CS_{events} and $F_{1\,max} = 0.9$. A classified event is regarded as a TP if the device was either (1) correctly classified (highlighted in green), (2) not learned at and was correctly classified as *unknown* (highlighted in light blue), or (3) classified as *unknown* prior to the time it was learned (column *unknown later learned*, highlighted in dark blue). Similarly, a FP of a certain appliance is an event classified as an event of the appliance which is actually originated from another appliance (represented by a column in the confusion matrix). Finally, a FN of an appliance is an event of the appliance which is recognized as the event of another appliance (represented by a row in the confusion matrix). It can be seen that appliances which have not been learned by the system (greyed out on the y-axis) are sometimes incorrectly classified as an already learned appliance (FP). Nevertheless, from a total number of 5713 events and ten appliances learned, the overall number of true positives adds up to 5586 ($FP = 170$ and $FN = 127$) indicating the potential of the concept.

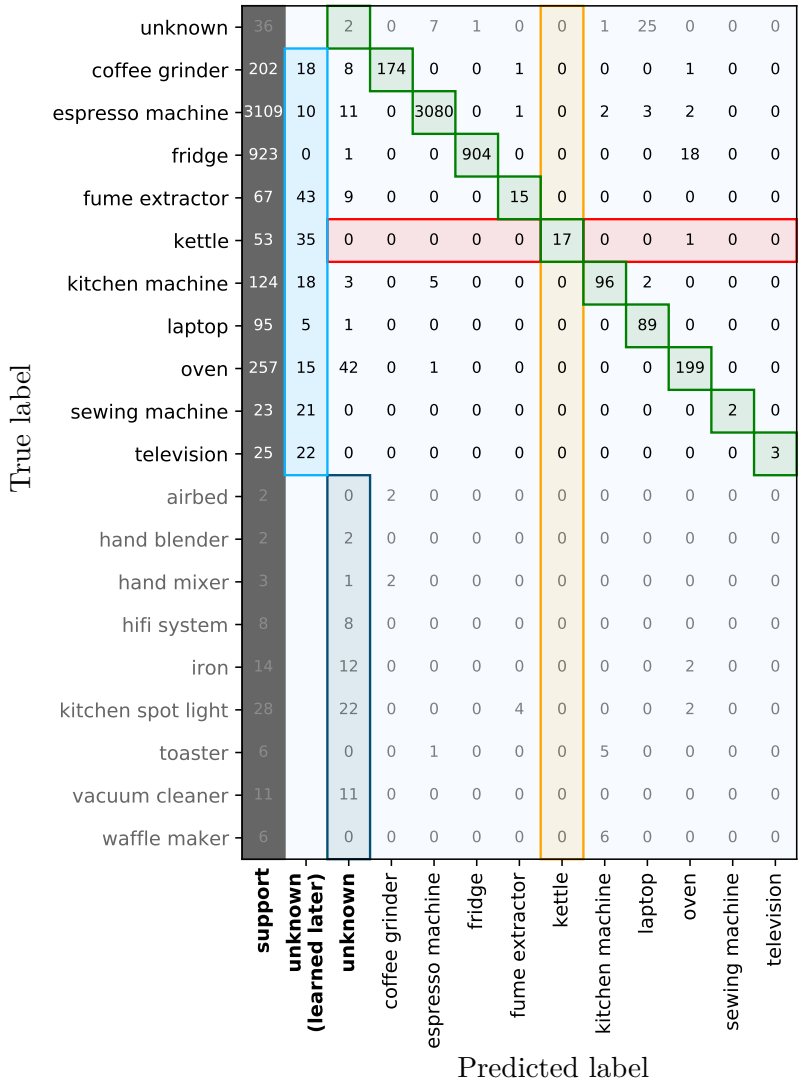

Figure 8.5: Confusion matrix of the MILM simulation with an SVM for $F_{1\ max} = 0.9$ and CS_{events}. Each matrix entry represents the number of particular predictions. FPs (orange) and FNs (red) are exemplarily shown for the *kettle*. All predictions regarded as TPs are: events of an appliance correctly assigned to it (green), appliances not learned (greyed out) predicted as unknown (dark blue), and at that time unknown appliances correctly predicted as unknown (light blue).

8.3.2 Discussion

The results of the MILM evaluation summarized in Table 8.4 are promising. An overall F_1-score of 92 % was reached (for $F_{1\ max} = 0.9$). While other supervised methods, such as the ones proposed by Kahl et al. [117] and Jorde et al. [133], reached perfect classification results on the laboratory WHITED dataset, typically, real system deployments, such as simulated by the conducted experiments, typically produce significantly worse results.

Remaining challenges of the proposed MILM approach still need to be addressed. The system was simulated only for events caused by a power increase (*up*-events). However, for *down*-events, the same cleaning process as applied to *up*-events can be applied. It is further argued that separate classifiers should be trained for *up* and *down* events if transient features are used, since the consumption over time is typically quite different for *up* and *down*. Furthermore, the concept does not yet include a load disaggregation into individual appliance load profiles. It rather delivers a set of classified appliance events with their corresponding timestamps from which appliance *on*-phases can be identified, assuming that down events are also recognized. Moreover, event-based NILM systems depend on the validity of the SCP. If events happen *simultaneously* (i.e., within a few milliseconds) like many events in the BLOND dataset, they might not be distinguishable. Still, in the residential domain, the SCP can assumed to be valid up to some extend (cf. [97]). Lastly, using plug meters to add appliances to the system assumes that each appliance of interest has a *plug*. While it can be argued that this holds for a lot of appliances, devices like a stove or most of the lighting are typically hard-wired. Those appliances could, however, still be learned by the system using the proposed cleaning method and a *classic* supervised NILM approach, in which the appliance is turned on and off multiple times during an explicit training phase. In the performed simulation, an average of 47 instances of the *POS* class (referring to appliance *up*-events) were included in the final training set (for CS_{events} and $F_{1\ max} = 0.9$). This reemphasizes that automating the ground truth generation process is required to minimize the time of training which requires explicit user interaction. The proposed MILM approach shows one way to achieve such an automation.

9 Conclusion and Future Work

The energy consumption of a home depends not only on the infrastructure, such as the insulation of the building and the energy efficiency of the appliance, but also on the behavior of the inhabitants. There is promising potential for energy savings by pinpointing users to unnecessary consumption or slightly adjusting their usage patterns without compromising comfort. NILM offers a retrospective way to provide individual appliance consumption data that helps to understand electricity usage and, therewith, to ultimately save energy. The research of NILM algorithms - especially for data-driven approaches - requires adequate datasets with corresponding ground truth data as well as methodologies to create more of such datasets when needed. Furthermore, most supervised NILM systems require a tedious training process that typically render their application impractical. The thesis at hand presented several strategies to solve existing challenges.

A set of design goals was defined for a system dedicated to generate universal electricity datasets that can be used to evaluate a wide variety of electricity-related algorithms. These goals have been implemented and evaluated for two different measurement systems that leverage hardware and software design methodologies to simultaneously measure time-synchronized, high-frequency voltage and current waveforms at aggregated-level and for multiple distributed appliances also at plug-level. The developed systems were integrated into a full stack recording framework. A central recording management system orchestrates the congregation and consistent storage of the data acquired by the individual measurement systems. The overall framework has proven itself and allowed a long-term data recording of over 100 days using 22 meters. While the employed techniques are tailored for electricity data, their incorporation to acquire other modalities such as water or gas consumption would require only minor adjustments.

To augment electricity data with ground truth labels, several post-conducted methods were investigated and a semi-automatic labeling algorithm was introduced. It is based on a probabilistic approach to identify appliance events and a clustering technique to add preliminary labels to each detected event. Further cleaning removes false positive events as well as multiple events which have been triggered by the same actual appliance event. The algorithm has been incorporated into an annotation tool which allows to conveniently supervise the automatic labeling.

The tool can increase the labeling efficiency up to 74 %.

The presented framework and labeling algorithm were used to collect and label the FIRED dataset. FIRED contains 101 days of electricity recordings of an apartment. The recordings include high-frequency aggregated (8 kHz) and individual (2 kHz) measurements of 21 different appliances as well as other modalities such as room temperature and appliance state changes. The data was further augmented with accurate and descriptive event labels of all appliances. The dataset and tools to interact with the data were made available to the research community to accelerate research of a wide variety of electricity-related algorithms.

To enhance the implementation of appliance recognition algorithms on resource constrained systems, such as smart meters, 27 features suggested by domain experts and four classifiers have been evaluated on four datasets. In the conducted experiments, active power achieved the best appliance recognition performance (F_1-score of 0.54) of any scalar feature, and WaveForm Approximation achieved the best performance (F_1-score of 0.88) of any multidimensional feature. Unsurprisingly, combining several of these features significantly improves performance. However, performance may also eventually degrade if too many features are added (aka *curse of dimensionality*). Therefore, the feature set $[P, cos\,\Phi, TRI, WFA]$ was proposed encompassing the time domain features WFA, P, and $cos\,\Phi$ as well as the frequency domain feature TRI in an overall feature vector of size 25. This vector is small enough not to suffer from the *curse of dimensionality*, comparatively easy to compute, and yet captures appliance specific characteristics to still achieve an F_1-score of 0.98 across all datasets. It has further been shown that a computationally fairly simple k-Nearest Neighbour classifier already achieves F_1-scores of up to 0.94.

The aggregated electrical energy consumption, as provided by smart meters, includes the consumption of all connected and active appliances. To recognize state changes of appliances (e.g., an appliance switch-on event) using aggregated-level data, several methods were investigated to clean the aggregated-level data from the contribution of other appliances. If cleaning is applied, the classification performance increases by up to 15 %. The best performances (F_1-scores of up to 0.98) were achieved by generating prototype waveforms which are subtracted from the post-event data in the time domain. Furthermore, the use of high-frequency plug-level recordings as additional training instances was investigated. This concept, denoted as *hybrid training*, has been proven to increase a system's classification performance by an additional 4 %, mainly due to data augmentation. Since supervised NILM methods still suffer from an intrusive training phase, the concept of Minimal-Intrusive Load Monitoring was proposed, which incorporates plug-level meters into a user-centric adaptive training phase. The plugs are used to collect training samples not disturbed by the energy consumption of other appliances, to

collect ground truth labels for the aggregated-level data, and to aid the overall training procedure in a straightforward way.

The NILM pipeline contains several stages that are typically evaluated individually by researchers. While a single step of this pipeline may perform well in an individual evaluation, it might perform worse in an evaluation encompassing all steps. It is conceivable that NILM research could benefit from a test-bed (similar to NILMTK [69]) which also supports high-frequency methods. As shown in this thesis, the combined use of plug-level data and aggregated-level data can improve NILM methods. Therefore, research may focus on the fusion of intrusive and non-intrusive load monitoring, especially given that smart home systems, which may already incorporate intrusive monitoring for some appliances, will become more prevalent in the near future.

Smart meters will employ techniques along NILM to enable consumer-centric services. Examples for these services are predictive maintenance, where the consumer is notified of the imminent failure of an appliance, or Ambient Assisted Living (AAL), where e.g., a stove, left on by mistake, is automatically switched off. Furthermore, user information inferred from appliance-level data can be used to optimally schedule the charging and usage of Energy Storage System (ESS) and Electric Vehicles (EVs) in combination with Renewable Energy Source (RES). This will help to reduce the electricity bill as well as the total energy consumption of a home. The key challenges to enable these services are: (1) Elimination of the vendor lock-in by providing standardized hardware interfaces, (real-time) transport protocols, and data formats; (2) Providing a dedicated ecosystem, like an *app store* for energy-based services [J21b, 172], since not all services apply to all users in the same way; (3) Protecting user privacy if data are not processed locally. In order for these services to gain widespread adoption, consumer interest must be aroused and kept constant. Since gamification has proven to facilitate that, research in specific gamification methods tailored to the energy sector should be further explored.

The contributions encompassed in this work will accelerate research in supervised NILM methods. Improving the performance and minimizing the intrusive training phase of NILM will ultimately help it to gain traction, especially as smart meters will be progressively installed in more homes in the upcoming years. As NILM provides electrical energy consumption at appliance-level, it benefits consumer services such as eco-feedback, predictive maintenance, or AAL as well as grid-operators alike. In addition to more convenience, this further helps us to save energy to minimize our environmental footprint.

List of Figures

List of Tables

***k*NN**	k-Nearest Neighbour
AAL	Ambient Assisted Living
AC	Alternating Current
ADC	Analog-to-Digital Converter
ANN	Artificial Neural Network
CF	Crest Factor
CNN	Convolutional Neural Network
COT	Current Over Time
CSV	Comma-Separated Values
CT	Current Transformer
CUSUM	Cumulative SUM
CV	Cross Validation
DAQ	Data Acquisition System
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DC	Direct Current
DNN	Deep Neural Network
DSP	Digital Signal Processor
DT	Decision Tree
EMI	Electromagnetic Interference
ESS	Energy Storage System
EV	Electric Vehicle
FF	Form Factor
FFT	Fast Fourier Transform
FHMM	Factorial Hidden Markov Model
FN	False Negatives
FP	False Positives
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GLR	General Likelihood Ratio
GOF	Goodness-of-Fit
GUI	Graphical User Interface
HAR	human activity recognition
HED	Harmonic Energy Distribution
HMM	Hidden Markov Model
HPO	Hyperparameter Optimization
IC	Integrated Circuit
ICR	Inrush Current Ratio
ILM	Intrusive Load Monitoring
IoT	Internet of Things

KFDA	Kernel Fisher Discriminant Analysis
KLM	Keystroke-Level Model
LAT	Log Attack Time
LLR	Log-Likelihood Ratio
MAE	Mean-Absolute Error
MAMI	Max-Min Ratio
MILM	Minimal-Intrusive Load Monitoring
MIR	Max-Inrush Ratio
MVR	Mean-Variance Ratio
NILM	Non-Intrusive Load Monitoring
NTP	Network Time Protocol
OER	Odd-Even Harmonics Ratio
PCB	Printed Circuit Board
PIR	Passive Infrared Sensor
PMR	Peak-Mean Ratio
PNR	Positive-Negative half cycle Ratio
PSS	Periods to Steady State
RES	Renewable Energy Source
RF	Random Forest
RMS	Root Mean Square
RMSE	Root-Mean-Square Error
RNN	Recurrent Neural Network
ROI	Region of Interest
RTC	Real Time Clock
RTT	Round-Trip Time
SC	Spectral Centroid
SCP	Switch Continuity Principle
SMPS	Switched-Mode Power Supply
SNR	Signal-to-Noise Ratio
SPF	Spectral Flatness
SPI	Serial Peripheral Interface
SVM	Support Vector Machine
TC	Temporal Centroid
THD	Total Harmonic Distortion
TN	True Negatives
TP	True Positives
TRI	Tristimulus
VIT	V-I Trajectory
WFA	WaveForm Approximation
WFD	Waveform Distortion

$\boldsymbol{cos}\,\Phi$	Phase Angle
$\boldsymbol{P}$	Active Power
$\boldsymbol{Q}$	Reactive Power
$\boldsymbol{R}$	Resistance
$\boldsymbol{S}$	Apparent Power
$\boldsymbol{Y}$	Admittance

Acknowledgments

This dissertation and the research contained herein were supervised by Professor Bernd Becker and were conducted at the Chair of Computer Architecture at the University of Freiburg.

I would first like to thank my supervisor, Professor Bernd Becker, whose expertise was invaluable in formulating the research questions and methodologies. Your valuable feedback, continuous support, and the allowed freedom for my research brought this work to a higher level.

Second, I like to thank Professor Kristof Van Laerhoven for agreeing to be the second examiner and Christoph Scholl for agreeing to be the third examiner of this dissertation. Furthermore, I would like to thank Professor Christian Schindelhauer for stepping in as an additional committee member and Professor Armin Biere for chairing the committee.

I would also like to thank my colleagues and friends, Dr. Philipp Scholl, Marc Pfeifer, Peter Winterer and Florian Wolling, for their help, time and creative ideas brought out during numerous discussions on the topic. Without them, this thesis would not have been possible.

In addition, I wish to thank all the people who have proofread this thesis in countless hours, especially Pascal Raiola. They pointed out weaknesses and highlighted where clarification was still needed. With their help numerous grammatical and spelling errors could be corrected.

I further want to thank my mother for her support and trust in me. Finally, thank you Sarah for your moral and energetic support. You and our little sunshine always cheered me up in the hard times of this work.

Author Contributions

[J21a] Benjamin Völker, Marc Pfeifer, Philipp M Scholl, and Bernd Becker. "A Framework to Generate and Label Datasets for Non-Intrusive Load Monitoring". In: *Energies*. Multidisciplinary Digital Publishing Institute, 2021, 14.1, p. 75.

[J21b] Benjamin Völker, Andreas Reinhardt, Anthony Faustine, and Lucas Pereira. "Watt's up at Home? Smart Meter Data Analytics from a Consumer-Centric Perspective". In: *Energies*. Multidisciplinary Digital Publishing Institute, 2021, 14.3, p. 719.

[C19a] Benjamin Völker, Philipp M Scholl, and Bernd Becker. "Semi-Automatic Generation and Labeling of Training Data for Non-Intrusive Load Monitoring". In: *Proceedings of the Tenth International Conference on Future Energy Systems*. e-Energy '19. ACM. Phoenix, USA, 2019.

[C19b] Benjamin Völker, Marc Pfeifer, Philipp M Scholl, and Bernd Becker. "A Versatile High Frequency Electricity Monitoring Framework for Our Future Connected Home". In: *International Conference on Sustainable Energy for Smart Cities*. SESC '19. Springer. Braga, Portugal, 2019, pp. 221–231.

[W20] Benjamin Völker, Philipp M. Pfeifer Marc amd Scholl, and Bernd Becker. "Annoticity: A Smart Annotation Tool and Data Browser for Electricity Datasets". In: *Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring*. Vol. 5. NILM '20. ACM. Virtual Event, Yokohama, Japan, 2020.

[N20] Benjamin Völker, Philipp M. Pfeifer Marc amd Scholl, and Bernd Becker. "FIRED: A Fully-labeled hIgh-fRequency Electricity Disaggregation Dataset". In: *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Built Environments*. Vol. 7. BuildSys '20. ACM. Virtual Event, Yokohama, Japan, 2020.

[PA21] Benjamin Völker, Marc Pfeifer, Florian Wolling, Philipp M Scholl, Josif Grabocka, and Bernd Becker. "Introducing MILM - A Hybrid Minimal-Intrusive Load Monitoring Approach". In: *Submitted to Proceedings of the Twelfth International Conference on Future Energy Systems*. e-Energy '21. ACM. Virtual Event, Torino, Italy, 2021.

[PA18] Benjamin Völker, Philipp M. Scholls, Tobias Schubert, and Bernd Becker. "Towards the Fusion of Intrusive and Non-intrusive Load Monitoring: A Hybrid Approach". In: *Proceedings of the Ninth International Conference on Future Energy Systems*. e-Energy '18. ACM. Karlsruhe, Germany, 2018, pp. 436–438.

[BC19] Philipp M Scholl, Benjamin Völker, Bernd Becker, and Kristof Van Laerhoven. "A multi-media exchange format for time-series dataset curation". In: *Human Activity Sensing*. Springer, 2019, pp. 111–119. ISBN: 978-3-030-13001-5. DOI: 10.1007/978-3-030-13001-5_8.

Author contributions currently under review and not yet published:

[N21] Benjamin Völker, Philipp M. Scholl, and Bernd Becker. "A Feature and Classifier Study for Appliance Event Classification". In: *Submitted to the 8th ACM International Conference on Systems for Energy-Efficient Built Environments*. Vol. 8. BuildSys '21. ACM. Coimbra, Portugal, 2021.

Additional author publications that did not contribute to this thesis:

[C21] Marc Pfeifer, Sebastian Böttcher, Sven Köhler, and Philipp M Scholl. "Teaching Embedded Systems by Constructing an Escape Room". In: *Proceedings of the 52nd Technical Symposium on Computer Science Education*. SIGCSE '21. ACM. Virtual Event, New York, USA, 2021.

[C17] Tobias Schubert, Benjamin Völker, Marc Pfeifer, and Bernd Becker. "The Smart MiniFab: An Industrial IoT Demonstrator Anywhere at Any Time". In: *International Conference on Smart Education and Smart E-Learning*. SEEL '17. Springer. Vilamoura, Portugal, 2017, pp. 253–262.

[C16] Benjamin Völker, Tobias Schubert, and Bernd Becker. "iHouse: A Voice-Controlled, Centralized, Retrospective Smart Home". In: *International Conference on Sensor Systems and Software*. SCUBE '16. Springer. Nice, France, 2016, pp. 68–80.

References

[1] Hannah Ritchie and Max Roser. "CO2 and Greenhouse Gas Emissions". In: *Our World in Data*. 2020. URL: https://ourworldindata.org/co2-and-other-greenhouse-gas-emissions (visited on 06/08/2021).

[2] International Energy Agency. "Key World Energy Statistics 2020". 2020. DOI: 10.1787/295f00f5-en.

[3] Eurostat - Statistics Explained. "Electricity Price Statistics". 2020. URL: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Electricity%5C%5Fprice%5C%5Fstatistics (visited on 06/08/2021).

[4] John K Dobson and JD Anthony Griffin. "Conservation Effect of Immediate Electricity Cost Feedback on Residential Consumption Behaviour". In: *Proceedings of the 7th ACEEE Summer Study on Energy Efficiency in Buildings*. Vol. 2. 1992.

[5] Anton Gustafsson and Magnus Gyllenswärd. "The Power-Aware Cord: Energy Awareness through Ambient Information Display". In: *Extended Abstracts on Human Factors in Computing Systems*. CHI '05. 2005.

[6] Tiffany Grace Holmes. "Eco-visualization: Combining Art and Technology to Reduce Energy Consumption". In: *Proceedings of the 6th SIGCHI Conference on Creativity & cognition*. ACM, 2007.

[7] Karen Ehrhardt-Martinez, Kat A Donnelly, and Skip Laitner. "Advanced Metering Initiatives and Residential Feedback Programs: A Meta-Review for Household Electricity-Saving Opportunities". In: *American Council for an Energy-Efficient Economy Washington, DC*. 2010.

[8] Jack Kelly and William Knottenbelt. "Does disaggregated electricity feedback reduce domestic electricity consumption? A systematic review of the literature". In: *arXiv preprint arXiv:1605.00962*. 2016.

[9] Tiago Serrenho, Paolo Zangheri, and Paolo Bertoldi. "Energy Feedback Systems: Evaluation of meta-studies on energy savings through feedback: Energy Efficiency Directive Articles 9–11 on feedback, billing and consumer information". In: *Publications Office of the European Union*. 2016. ISSN: 1831-9424. DOI: 10.2790/565532.

[10] Statistisches Bundesamt. “Stromverbrauch der privaten Haushalte nach Haushaltsgrößenklassen”. Aug. 2020. URL: https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Umwelt/UGR/private-haushalte/Tabellen/stromverbrauch-haushalte.html (visited on 06/14/2021).

[11] Discovergy GmbH. “Discover your energy”. URL: https://discovergy.com (visited on 06/08/2021).

[12] sense. “sense - Take command of your energy use with total home monitoring”. URL: https://sense.com (visited on 06/08/2021).

[13] Google. “Google PowerMeter”. discontinued in †2011. 2009. URL: https://sites.google.com/site/powermeterpartners/google-meter-api (visited on 07/16/2021).

[14] Microsoft Corp. “Microsoft Hohm”. discontinued in †2012. 2009. URL: https://news.microsoft.com/2009/06/24/microsoft-hohm-helps-consumers-save-money-and-energy/ (visited on 07/16/2021).

[15] A. Reinhardt and C. Klemenjak. “How does Load Disaggregation Performance Depend on Data Characteristics? Insights from a Benchmarking Study”. In: *Proceedings of the 11th International Conference on Future Energy Systems*. e-Energy ’20. ACM, 2020.

[16] Stephen Makonin, Fred Popowich, Lyn Bartram, Bob Gill, and Ivan V. Bajić. “AMPds: A Public Dataset for Load Disaggregation and Eco-Feedback Research”. In: *Proceeding of the Annual Electrical Power and Energy Conference*. EPEC ’13. 2013. ISBN: 978-1-4799-0106-7. DOI: 10.1109/EPEC.2013.6802949.

[17] J Zico Kolter and Matthew J Johnson. “REDD : A Public Data Set for Energy Disaggregation Research”. In: *Proceedings of the 1st KDD Workshop on Data Mining Applications in Sustainability*. SustKDD ’12. 2011. ISBN: 978-1-4503-0840-3.

[18] Markus Weiss, Adrian Helfenstein, Friedemann Mattern, and Thorsten Staake. “Leveraging smart meter data to recognize home appliances”. In: *International Conference on Pervasive Computing and Communications*. IEEE, 2012.

[19] Matthias Kahl, Thomas Kriechbaumer, Anwar Ul Haq, and Hans-Arno Jacobsen. “Appliance Classification Across Multiple High Frequency Energy Datasets”. In: *International Conference on Smart Grid Communications*. SmartGridComm ’17. IEEE, 2017.

[20] Yassine Himeur, Abdullah Alsalemi, Faycal Bensaali, and Abbes Amira. "Robust event-based non-intrusive appliance recognition using multi-scale wavelet packet tree and ensemble bagging tree". In: *Applied Energy*. Elsevier, 2020, 267.

[21] Thomas Kriechbaumer and Hans-Arno Jacobsen. "BLOND, a building-level office environment dataset of typical electrical appliances". In: *Scientific data*. Nature Publishing Group, 2018, 5.

[22] Matthias Kahl, Anwar Ul Haq, Thomas Kriechbaumer, and Hans-Arno Jacobsen. "WHITED - A Worldwide Household and Industry Transient Energy Data Set". In: *3rd International Workshop on Non-Intrusive Load Monitoring*. NILM '16. 2016.

[23] Jingkun Gao, Suman Giri, Emre Can Kara, and Mario Bergés. "PLAID: a Public Dataset of High-Resolution Electrical Appliance Measurements for Load Identification Research". In: *Proceedings of the 1st Conference on Embedded Systems for Energy-Efficient Buildings*. BuildSys '14. ACM, 2014.

[24] Werner Siemens. "Zur Geschichte der dynamo-elektrischen Maschine". In: *Wissenschaftliche und Technische Arbeiten: Zweiter Band. Technische Arbeiten*. Springer Berlin Heidelberg, 1891. ISBN: 978-3-642-64936-3. DOI: 10.1007/978-3-642-64936-3_68.

[25] Jacopo Torriti. "Appraising the Economics of Smart Meters: Costs and Benefits". Routledge, 2020.

[26] N. Uribe-Pérez, L. Hernández, D. De la Vega, and I. Angulo. "State of the Art and Trends Review of Smart Metering in Electricity Grids". In: *Applied Sciences*. 2016, 6.68.

[27] K. Förderer, M. Lösch, R. Növer, M. Ronczka, and H. Schmeck. "Smart Meter Gateways: Options for a BSI-Compliant Integration of Energy Management Systems". In: *Applied Sciences*. 2019, 9.1634.

[28] Y. Wang, Q. Chen, T. Hong, and C. Kang. "Review of Smart Meter Data Analytics: Applications, Methodologies, and Challenges". In: *Transactions on Smart Grid*. IEEE, 2019, 10.3.

[29] Studiosus Kirchhoff. "Über den Durchgang eines elektrischen Stromes durch eine Ebene, insbesondere durch eine kreisförmige". In: *Annalen der Physik*. Wiley Online Library, 1845, 140.4.

[30] Patrick Huber, Alberto Calatroni, Andreas Rumsch, and Andrew Paice. "Review on Deep Neural Networks Applied to Low-Frequency NILM". In: *Energies*. 2021, 14.9. ISSN: 1996-1073. DOI: 10.3390/en14092390.

[31] Jana Clement, Joern Ploennigs, and Klaus Kabitzsch. “Smart Meter: Detect and Individualize ADLs”. In: *Ambient Assisted Living*. Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-27491-6_8.

[32] C. Chalmers, P. Fergus, C. A. Curbelo Montanez, S. Sikdar, F. Ball, et al. “Detecting Activities of Daily Living and Routine Behaviours in Dementia Patients Living Alone Using Smart Meter Load Disaggregation”. In: *Transactions on Emerging Topics in Computing*. IEEE, 2020. DOI: 10.1109/TETC.2020.2993177.

[33] H. Bousbiat, C. Klemenjak, G. Leitner, and W. Elmenreich. “Augmenting an Assisted Living Lab with Non-Intrusive Load Monitoring”. In: *International Instrumentation and Measurement Technology Conference*. I2MTC ’20. 2020. DOI: 10.1109/I2MTC43012.2020.9128406.

[34] Andreas Reinhardt and Christoph Klemenjak. “Device-Free User Activity Detection using Non-Intrusive Load Monitoring: A Case Study”. In: *Proceedings of the 2nd Workshop on Device Free Human Sensing*. DFHS ’20. ACM, 2020.

[35] Ahmed Zoha, Alexander Gluhak, Muhammad Ali Imran, and Sutharshan Rajasegarar. “Non-Intrusive Load Monitoring Approaches for Disaggregated Energy Sensing: A Survey”. In: *Sensors*. MDPI, 2012, 12.12.

[36] Anthony Faustine, Nerey Henry Mvungi, Shubi Kaijage, and Kisangiri Michael. “A Survey on Non-Intrusive Load Monitoring Methodies and Techniques for Energy Disaggregation Problem”. In: *arXiv preprint: 1703.00785*. 2017.

[37] José Alcalá, Jesús Ureña, Álvaro Hernández, and David Gualda. “Event-Based Energy Disaggregation Algorithm for Activity Monitoring From a Single-Point Sensor”. In: *Transactions on Instrumentation and Measurement*. IEEE, 2017, 66.10.

[38] Karim Said Barsim, Roman Streubel, and Bin Yang. “Unsupervised Adaptive Event Detection for Building-Level Energy Disaggregation”. In: *Proceedings of Power and Energy Student Summt, Stuttgart, Germany*. 2014. PESS ’14.

[39] Lucas Pereira. “Developing and Evaluating a Probabilistic Event Detector for Non-Intrusive Load Monitoring”. In: *Sustainable Internet and ICT for Sustainability*. SustainIT ’17. IEEE, 2017.

[40] Mohamed Nait Meziane, Philippe Ravier, Guy Lamarque, Jean-Charles Le Bunetel, and Yves Raingeaud. “High Accuracy Event Detection for Non-Intrusive Load Monitoring”. In: *International Conference on Acoustics, Speech and Signal Processing*. ICASSP ’17. IEEE, 2017.

[41] Awet Abraha Girmay and Christian Camarda. “Simple Event Detection and Disaggregation Approach for Residential Energy Estimation”. In: *Proceedings of the 3rd International Workshop on Non-Intrusive Load Monitoring (NILM)*. NILM ’16. 2016.

[42] Houda Ben Attia Sethom, Sarra Houidi, François Auger, Houda Ben, Attia Sethom, et al. “Multivariate Event Detection Methods for Non-Intrusive Load Monitoring in Smart Homes and Residential Buildings”. In: *Energy and Buildings*. 2019, 208.

[43] H. Kim, M. Marwah, M. F. Arlitt, G. Lyon, and J. Han. “Unsupervised Disaggregation of Low Frequency Power Measurements”. In: *Proceedings of the 11th SIAM International Conference on Data Mining*. 2011. DOI: 10.1137/1.9781611972818.64.

[44] J Zico Kolter and Tommi Jaakkola. “Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation”. In: *Artificial intelligence and statistics*. PMLR, 2012.

[45] Oliver Parson, S Ghosh, M Weal, and A Rogers. “Non-intrusive Load Monitoring using Prior Models of General Appliance Types”. In: *Proceeding of the 26th AAAI Conference on Artificial Intelligence*. AAAI ’12. 2012.

[46] Stephen Makonin, Fred Popowich, Ivan V. Bajic, Bob Gill, and Lyn Bartram. “Exploiting HMM Sparsity to Perform Online Real-Time Nonintrusive Load Monitoring”. In: *Transactions on Smart Grid*. IEEE, 2015. ISSN: 19493053. DOI: 10.1109/TSG.2015.2494592.

[47] Christian Beckel, Wilhelm Kleiminger, Romano Cicchetti, Thorsten Staake, and Silvia Santini. “The ECO Data Set and the Performance of Non-Intrusive Load Monitoring Algorithms”. In: *Proceedings of the 1st Conference on Embedded Systems for Energy-Efficient Buildings*. BuildSys ’14. ACM, 2014.

[48] Chaoyun Zhang, Mingjun Zhong, Zongzuo Wang, Nigel Goddard, and Charles Sutton. “Sequence-to-Point Learning with Neural Networks for Non-Intrusive Load Monitoring”. In: *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. AAAI ’18. 2018.

[49] Jin-Gyeom Kim and Bowon Lee. “Appliance Classification by Power Signal Analysis Based on Multi-Feature Combination Multi-Layer LSTM”. In: *Energies*. MDPI, 2019, 12.14.

[50] Anthony Faustine, Lucas Pereira, Hafsa Bousbiat, and Shridhar Kulkarni. "UNet-NILM: A Deep Neural Network for Multi-tasks Appliances State Detection and Power Estimation in NILM". In: *Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring*. NILM '20. 2020.

[51] Eduardo Gomes and Lucas Pereira. "PB-NILM: Pinball Guided Deep Non-Intrusive Load Monitoring". In: *Access*. IEEE, 2020, 8.

[52] Kyle D Anderson, Mario E Bergés, Adrian Ocneanu, Diego Benitez, and Jose MF Moura. "Event Detection for Non Intrusive Load Monitoring". In: *38th Annual Conference on Industrial Electronics Society*. IECON '12. IEEE, 2012.

[53] K. C. Armel, A. Gupta, G. Shrimali, and A. Albert. "Is disaggregation the holy grail of energy efficiency? The case of electricity". In: *Energy Policy*. Elsevier, 2013, 52.

[54] Elton Law. "Impyute - A library of missing data imputation algorithms written in Python 3". 2017. URL: `http://impyute.readthedocs.io` (visited on 06/08/2021).

[55] Jesús Díaz García, Pere Brunet Crosa, Isabel Navazo Álvaro, and Pere Pau Vázquez Alcocer. "Downsampling Methods for Medical Datasets". In: *Proceedings of the International conferences Computer Graphics, Visualization, Computer Vision and Image Processing 2017*. IADIS Press, 2017.

[56] Andreas Reinhardt, Frank Englert, and Delphine Christin. "Averting the privacy risks of smart metering by local data preprocessing". In: *Pervasive and Mobile Computing*. Elsevier, 2015, 16.

[57] Rithwik Kukunuri, Nipun Batra, and Hongning Wang. "An Open Problem: Energy Data Super-Resolution". In: *Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring*. 2020.

[58] Kitisak Osathanunkul and Khukrit Osathanunkul. "Different Sampling Rates on Neural NILM Energy Disaggregation". In: *Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering*. IEEE, 2019.

[59] Bohao Huang, Mary Knox, Kyle Bradbury, Leslie M Collins, and Richard G Newell. "Non-intrusive load monitoring system performance over a range of low frequency sampling rates". In: *6th International Conference on Renewable Energy Research and Applications*. ICRERA '17. IEEE, 2017.

[60] Jana Huchtkoetter and Andreas Reinhardt. “On the Impact of Temporal Data Resolution on the Accuracy of Non-Intrusive Load Monitoring”. In: *Proceedings of the 7th International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. BuildSys ’20. ACM, 2020.

[61] Jana Huchtkoetter and Andreas Reinhardt. “A study on the impact of data sampling rates on load signature event detection”. In: *Energy Informatics*. Springer, 2019, 2.1.

[62] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc., 2012.

[63] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. “Multi-column Deep Neural Networks for Image Classification”. In: *Conference on Computer Vision and Pattern Recognition*. 2012. DOI: 10.1109/CVPR.2012.6248110.

[64] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. “ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning”. In: *International Joint Conference on Neural Networks (World Congress on Computational Intelligence)*. IEEE, 2008.

[65] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. “SMOTE: Synthetic Minority Over-Sampling Technique”. In: *Journal of Artificial Intelligence Research*. 2002, 16.

[66] Aida Ali, Siti Mariyam Shamsuddin, and Anca L Ralescu. “Classification with class imbalance problem”. In: *International Journal of Advances in Soft Computing and its Applications*. 2013, 5.3.

[67] XGBoost Developers. *XGBoost Documentation - Parameters*. 2020. URL: https://xgboost.readthedocs.io/en/latest/parameter.html (visited on 07/12/2021).

[68] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, et al. “Efficient and Robust Automated Machine Learning”. In: *Advances in Neural Information Processing Systems 28*. Curran Associates, Inc., 2015.

[69] Nipun Batra, Jack Kelly, Oliver Parson, Haimonti Dutta, William Knottenbelt, et al. “NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring”. In: *Proceedings of the 5th international conference on Future energy systems*. e-Energy ’14. ACM, 2014.

[70] Jack Kelly and William Knottenbelt. "Neural NILM: Deep Neural Networks Applied to Energy Disaggregation". In: *Proceedings of the 2nd International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 2015.

[71] Pico Technology. *TA041 - 25 MHz 700 V Differential Probe - User's Manual*. URL: https://www.picotech.com/download/manuals/ta041-differential-probe-users-guide.pdf (visited on 06/08/2021).

[72] *Datasheet NI-9239*. National Instruments. URL: https://www.ni.com/pdf/manuals/375939b%5C%5F02.pdf (visited on 06/08/2021).

[73] Enmetric. "The Enmetric Power Port". URL: https://leeduser.buildinggreen.com/sites/default/files/credit%5C%5Fdocumentation/Enmetric%5C%20Product%5C%20Brochure.pdf (visited on 06/08/2021).

[74] PowerhouseDynamics. "Equipment and energy management for Retail Chains, Health Clinics, and other Commercial Facilities". URL: http://www.powerhousedynamics.com (visited on 06/08/2021).

[75] Jack Kelly and William Knottenbelt. "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes". In: *Scientific data*. Nature Publishing Group, 2015, 2.

[76] EDF Energy. "Your home energy monitor - Step-by-step user guide". URL: https://www.edfenergy.com/sites/default/files/r1089%5C%5Flg%5C%5Fcug%5C%5Fv2%5C%5Faw5%5C%5Fe13.pdf (visited on 06/08/2021).

[77] Current Cost. "Energy Transmitter". URL: http://currentcost.com/product-transmitter.html (visited on 06/08/2021).

[78] Landis+Gyr. *Landis+Gyr E750*. URL: https://www.landisgyr.de/product/landisgyr-e750/ (visited on 06/08/2021).

[79] plugwise. "Plugwise Plug - Measure the energy use per device with the Plugwise Plug". URL: https://www.plugwise.com/product/plug/?lang=en (visited on 06/08/2021).

[80] Anwar Ul Haq, Thomas Kriechbaumer, Matthias Kahl, and Hans Arno Jacobsen. "CLEAR - A Circuit Level Electric Appliance Radar for the Electric Cabinet". In: *Proceedings of the International Conference on Industrial Technology*. IEEE, 2017. DOI: 10.1109/ICIT.2017.7915521.

[81] Thomas Kriechbaumer, Anwar Ul Haq, Matthias Kahl, and Hans Arno Jacobsen. "MEDAL: A Cost-Effective High-Frequency Energy Data Acquisition System for Electrical Appliances". In: *Proceedings of the 8th International Conference on Future Energy Systems*. 2017. e-Energy '17. DOI: 10.1145/3077839.3077844.

[82] LEM. *Current Transducer HAL 50 .. 600S*. URL: https://www.lem.com/sites/default/files/products%5C%5Fdatasheets/hal%5C%5F50%5C%5F600-s.pdf (visited on 06/08/2021).

[83] BLOCK. *VB 3,2/1/6 Sicherheitstransformator*. URL: https://www.block.eu/de%5C%5FDE/produktvariante/vb-3216/ (visited on 06/08/2021).

[84] Analog Devices. *50 kSPS, 6-Channel, Simultaneous Sampling, Bipolar 16-Bit ADC*. URL: https://www.analog.com/media/en/technical-documentation/data-sheets/AD7656A.pdf (visited on 06/08/2021).

[85] Lattice Semiconductor. *MachXO2 - Flexible Interface Bridging FPGA*. URL: https://www.latticesemi.com/en/Products/FPGAandCPLD/MachXO2 (visited on 06/08/2021).

[86] LattePanda. *LattePanda - A windows 10 development board for everything*. URL: https://www.lattepanda.com (visited on 06/08/2021).

[87] Microchip Technology Inc. *2.7V 12-Bit A/D Converter with SPI Serial Interface*. URL: http://ww1.microchip.com/downloads/en/DeviceDoc/21290F.pdf (visited on 06/08/2021).

[88] Microchip Technology Inc. *8-bit Atmel Microcontroller with 16/32/64/128K Bytes In-System Programmable Flash*. URL: http://ww1.microchip.com/downloads/en/devicedoc/atmel-8272-8-bit-avr-microcontroller-atmega164a%5C%5Fpa-324a%5C%5Fpa-644a%5C%5Fpa-1284%5C%5Fp%5C%5Fdatasheet.pdf (visited on 06/08/2021).

[89] RASPBERRY PI FOUNDATION. *Raspberry Pi 3 - Model B+*. URL: https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf (visited on 06/08/2021).

[90] Oliver Parson, Grant Fisher, April Hersey, Nipun Batra, Jack Kelly, et al. "Dataport and NILMTK: A Building Data Set Designed for Non-intrusive Load Monitoring". In: *Global Conference on Signal and Information Processing*. GlobalSIP '15. IEEE, 2015.

[91] Nipun Batra, Manoj Gulati, Amarjeet Singh, and Mani B Srivastava. "It's Different: Insights into home energy consumption in India". In: *Proceedings of the 5th Workshop on Embedded Systems For Energy-Efficient Buildings*. ACM, 2013.

[92] Stephen Makonin, Z Jane Wang, and Chris Tumpach. "RAE: The Rainforest Automation Energy Dataset for Smart Grid Meter Data Analysis". In: *data*. MDPI, 2018, 3.1.

[93] Miguel Ribeiro, Lucas Pereira, Filipe Quintal, and Nuno Nunes. "SustDataED: A Public Dataset for Electric Energy Disaggregation Research". In: *Proceedings of ICT for Sustainability*. ICT4S '16. Atlantis Press, Aug. 2016. DOI: 10.2991/ict4s-16.2016.36.

[94] Kyle Anderson, Adrian Filip Ocneanu, Diego Benitez, Derrick Carlson, Anthony Rowe, et al. "BLUED : A Fully Labeled Public Dataset for Event-Based Non-Intrusive Load Monitoring Research". In: *Proceeding of the 2nd KDD Workshop on Data Mining Applications in Sustainability*. SustKDD '12. 2012. ISBN: 978-1-4503-1558-6.

[95] DENT Instruments. *User's manual PowerScout 3, PowerScout 18, ViewPoint*. URL: https://www.dentinstruments.com/hubfs/powerscout3-powerscout18-user-manual.pdf (visited on 06/08/2021).

[96] George William Hart. "Nonintrusive Appliance Load Monitoring". In: *Proceedings of the IEEE*. IEEE, 1992, 80.12.

[97] Stephen Makonin. "Investigating the Switch Continuity Principle Assumed in Non-Intrusive Load Monitoring (NILM)". In: *Conference on Electrical and Computer Engineering*. CCECE '16. IEEE, 2016.

[98] Lucas Pereira and Nuno Nunes. "An empirical exploration of performance metrics for event detection algorithms in Non-Intrusive Load Monitoring". In: *Sustainable Cities and Society*. Elsevier, 2020, 62.

[99] Lucas Pereira and Nuno J Nunes. "Semi-Automatic Labeling for Public Non-Intrusive Load Monitoring Datasets". In: *Sustainable Internet and ICT for Sustainability*. SustainIT '15. IEEE, 2015.

[100] Matthias Kahl, Thomas Kriechbaumer, Daniel Jorde, Anwar Ul Haq, and Hans-Arno Jacobsen. "Appliance Event Detection - A Multivariate, Supervised Classification Approach". In: *Proceedings of the 10th International Conference on Future Energy Systems*. e-Energy '20. ACM, 2019.

[101] Lucas Pereira, Miguel Ribeiro, and Nuno Nunes. "Engineering and Deploying a Hardware and Software Platform to Collect and Label Non-Intrusive Load Monitoring Datasets". In: *Sustainable Internet and ICT for Sustainability*. SustainIT '17. IEEE, 2017.

[102] Paula Meehan, Conor McArdle, and Stephen Daniels. “An Efficient, Scalable Time-Frequency Method for Tracking Energy Usage of Domestic Appliances Using a Two-Step Classification Algorithm”. In: *Energies*. MDPI, 2014, 7.11.

[103] Dong Luo, Leslie K Norford, Steven R Shaw, and Steven B Leeb. “Monitoring HVAC Equipment Electrical Loads from a Centralized Location – Methods and Field Test Results”. In: *ASHRAE Transactions*. American Society of Heating, Refrigeration and Air Conditioning Engineers, Inc., 2002, 108.

[104] Kien Nguyen Trung, Eric Dekneuvel, Benjamin Nicolle, Olivier Zammit, Cuong Nguyen Van, et al. “Event Detection and Disaggregation Algorithms for NIALM System”. In: *Proceedings of 2nd International Non-Intrusive Load Monitoring Workshop*. NILM ’14. 2014.

[105] Leen De Baets, Joeri Ruyssinck, Dirk Deschrijver, and Tom Dhaene. “Event Detection in NILM using Cepstrum smoothing”. In: *3rd International Workshop on Non-Intrusive Load Monitoring*. NILM ’16. 2016.

[106] Robert Cox, Steven B Leeb, Steven R Shaw, and Leslie K Norford. “Transient Event Detection for Nonintrusive Load Monitoring and Demand Side Management Using Voltage Distortion”. In: *21st Annual Conference on Applied Power Electronics Conference and Exposition*. APEC ’06. IEEE, 2006.

[107] Yuanwei Jin, Eniye Tebekaemi, Mario Berges, and Lucio Soibelman. “Robust Adaptive Eevent Detection in Non-Intrusive Load Monitoring for Energy Aware Smart Facilities”. In: *International Conference on Acoustics, Speech and Signal Processing*. ICASSP ’11. IEEE, 2011.

[108] Matthias Kahl. “Machine Learning for Non-Intrusive Load Monitoring”. PhD thesis. Technische Universität München, 2019.

[109] Benjamin Wild, Karim Said Barsim, and Bin Yang. “A new Unsupervised Event Detector for Non-Intrusive Load Monitoring”. In: *Global Conference on Signal and Information Processing*. GlobalSIP ’15. IEEE, 2015.

[110] Michele Basseville and Igor V Nikiforov. “Detection of Abrupt Changes: Theory and Application”. Vol. 104. Pprentice Hall Englewood Cliffs, 1993.

[111] Mario Berges, Ethan Goldman, H Scott Matthews, Lucio Soibelman, and Kyle Anderson. “User-Centered Nonintrusive Electricity Load Monitoring for Residential Buildings”. In: *Journal of computing in civil engineering*. American Society of Civil Engineers, 2011, 25.6.

[112] Hader AD Azzini, Ricardo Torquato, and Luiz CP da Silva. "Event Detection Methods for Nonintrusive Load Monitoring". In: *PES General Meeting, Conference & Exposition*. IEEE, 2014.

[113] Hanen Berriri, Mohamed Wissem Naouar, and Ilhem Slama-Belkhodja. "Easy and Fast Sensor Fault Detection and Isolation Algorithm for Electrical Drives". In: *Transactions on Power Electronics*. IEEE, 2011, 27.2.

[114] Steven B Leeb, Steven R Shaw, and James L Kirtley. "Transient Event Detection in Spectral Envelope Estimates for Nonintrusive Load Monitoring". In: *Transactions on Power Delivery*. IEEE, 1995, 10.3.

[115] Zhuang Zheng, Hainan Chen, and Xiaowei Luo. "A Supervised Event-Based Non-Intrusive Load Monitoring for Non-Linear Appliances". In: *Sustainability*. MDPI, 2018, 10.4.

[116] Daniel Jorde, Matthias Kahl, and Hans-Arno Jacobsen. "MEED: An Unsupervised Multi-Environment Event Detector for Non-Intrusive Load Monitoring". In: *International Conference on Communications, Control, and Computing Technologies for Smart Grids*. SmartGridComm '19. IEEE, 2019.

[117] Matthias Kahl, Anwar Ul Haq, Thomas Kriechbaumer, and Hans-Arno Jacobsen. "A Comprehensive Feature Study for Appliance Recognition on High Frequency Energy Data". In: *Proceedings of the 8th International Conference on Future Energy Systems*. e-Energy '17. Shatin, Hong Kong: Association for Computing Machinery, 2017. ISBN: 978-1-4503-5036-5. DOI: 10.1145/3077839.3077845.

[118] Sarra Houidi, François Auger, Houda Ben Attia Sethom, Laurence Miègeville, Dominique Fourer, et al. "Statistical Assessment of Abrupt Change Detectors for Non-Intrusive Load Monitoring". In: *International Conference on Industrial Technology*. ICIT '18. IEEE, 2018.

[119] Anwar Ul Haq. "Appliance Event Detection for Non-Intrusive Load Monitoring in Complex Environments". PhD thesis. Technische Universität München, 2018.

[120] WL Chan, Albert TP So, and LL Lai. "Harmonics Load Signature Recognition by Wavelets Transform". In: *International Conference on Electric Utility Deregulation and Restructuring and Power Technologies*. DRPT '00. IEEE, 2000.

[121] Hong Yin Lam, GSK Fung, and WK Lee. "A Novel Method to Construct Taxonomy of Electrical Appliances Based on Load Signatures". In: *Transactions on Consumer Electronics*. IEEE, 2007, 53.2.

[122] Taha Hassan, Fahad Javed, and Naveed Arshad. “An Empirical Investigation of V-I Trajectory Based Load Signatures for Non-Intrusive Load Monitoring”. In: *Transactions on Smart Grid*. IEEE, 2013, 5.2.

[123] A Longjun Wang, B Xiaomin Chen, C Gang Wang, and D Hua. “Non-intrusive load monitoring algorithm based on features of V–I trajectory”. In: *Electric Power Systems Research*. Elsevier, 2018, 157.

[124] Hong-Tzer Yang, Hsueh-Hsien Chang, and Ching-Lung Lin. “Load Identification in Neural Networks for a Non-intrusive Monitoring of Industrial Electrical Loads”. In: *11th International Conference on Computer Supported Cooperative Work in Design*. IEEE, 2007.

[125] Sidhant Gupta, Matthew S Reynolds, and Shwetak N Patel. “ElectriSense: Single-Point Sensing Using EMI for Electrical Event Detection and Classification in the Home”. In: *Proceedings of the 12th International Conference on Ubiquitous Computing*. ACM, 2010.

[126] J. Gao, E. C. Kara, S. Giri, and M. Bergés. “A feasibility study of automated plug-load identification from high-frequency measurements”. In: *Global Conference on Signal and Information Processing*. GlobalSIP ’15. IEEE, 2015.

[127] Leen De Baets, Joeri Ruyssinck, Chris Develder, Tom Dhaene, and Dirk Deschrijver. “Appliance classification using VI trajectories and convolutional neural networks”. In: *Energy and Buildings*. Elsevier, 2018, 158.

[128] Jian Liang, Simon KK Ng, Gail Kendall, and John WM Cheng. “Load Signature Study—Part I: Basic Concept, Structure, and Methodology”. In: *Transactions on Power Delivery*. IEEE, 2010, 25.2.

[129] Nasrin Sadeghianpourhamami, Joeri Ruyssinck, Dirk Deschrijver, Tom Dhaene, and Chris Develder. “Comprehensive feature selection for appliance classification in NILM”. In: *Energy and Buildings*. Elsevier, 2017, 151.

[130] Peter Davies, Jon Dennis, Jack Hansom, William Martin, Aistis Stankevicius, et al. “Deep Neural Networks for Appliance Transient Classification”. In: *International Conference on Acoustics, Speech and Signal Processing*. ICASSP ’19. IEEE, 2019.

[131] Jingkun Gao, Emre Can Kara, Suman Giri, and Mario Bergés. “A feasibility study of automated plug-load identification from high-frequency measurements”. In: *Global Conference on Signal and Information Processing*. GlobalSIP ’15. IEEE, 2015.

[132] Chuan Choong Yang, Chit Siang Soh, and Vooi Voon Yap. "A systematic approach in appliance disaggregation using k-nearest neighbours and naive Bayes classifiers for energy efficiency". In: *Energy Efficiency*. Springer, 2018, 11.1.

[133] Daniel Jorde, Thomas Kriechbaumer, and Hans-Arno Jacobsen. "Electrical Appliance Classification using Deep Convolutional Neural Networks on High Frequency Current Measurements". In: *International Conference on Communications, Control, and Computing Technologies for Smart Grids*. SmartGridComm '18. IEEE, 2018.

[134] George W Hart. "Prototype Nonintrusive Appliance Load Monitor". In: *MIT Energy Laboratory Technical Report, and Electric Power Research Institute Technical Report, September 1985*. 1985.

[135] Mingjun Zhong, Nigel Goddard, and Charles Sutton. "Signal Aggregate Constraints in Additive Factorial HMMs, with Application to Energy Disaggregation". In: *Advances in Neural Information Processing Systems*. 2014.

[136] Oliver Kramer, Thole Klingenberg, Michael Sonnenschein, and Olaf Wilken. "Non-intrusive appliance load monitoring with bagging classifiers". In: *Logic Journal of the IGPL*. Oxford University Press, 2015, 23.3.

[137] Hông-Ân Cao, Tri Kurniawan Wijaya, Karl Aberer, and Nuno Nunes. "A Collaborative Framework for Annotating Energy Datasets". In: *International Conference on Big Data*. IEEE, 2015.

[138] Peter Faymonville, Kai Wang, John Miller, et al. "CAPTCHA-based Image Labeling on the Soylent Grid". In: *Proceedings of the SIGKDD Workshop on Human Computation*. ACM, 2009.

[139] Jana Huchtkoetter, Andreas Reinhardt, and Sakif Hossain. "ANNO: A Time Series Annotation Tool to Evaluate Event Detection Algorithms". In: *International Workshop on Simulation Science*. Springer, 2019.

[140] DIN EN 60664-1 VDE 0110-1. "Isolationskoordination für Niederspannungsbetriebsmittel - Teil 2–1: Anwendungsrichtlinie - Erläuterungen zur Anwendung der Normenreihe IEC 60664, Bemessungsbeispiele und Isolationsprüfungen". Norm. (IEC/TR 60664-2-1:2011 + Cor.:2011). June 2012.

[141] *ADE9000 - High Performance, Multiphase Energy, and Power Quality Monitoring IC*. ADE9000. Rev. A. Analog Devices. URL: `www.analog.com/media/en/technical-documentation/data-sheets/ADE9000.pdf` (visited on 06/08/2021).

[142] DIN VDE 0100-443. “DIN VDE 0100-443 VDE 0100-443:2016-10 - Errichten von Niederspannungsanlagen Teil 4-44: Schutzmaßnahmen - Schutz bei Störspannungen und elektromagnetischen Störgrößen - Abschnitt 443: Schutz bei transienten Überspannungen infolge atmosphärischer Einflüsse oder von Schaltvorgängen”. Norm. (IEC 60364-4-44:2007/A1:2015, modifiziert); Deutsche Übernahme HD 60364-4-443:2016. Oct. 2016.

[143] *Split core current transformer - SCT013-000*. SCT013. YHDC. URL: http://en.yhdc.com/product/SCT013-401.html (visited on 06/08/2021).

[144] *ESP32 Hardware Reference*. Espressif. URL: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/index.html (visited on 06/09/2021).

[145] *DS3231 - Extremely Accurate I2C-Integrated*. Maxim Integrated. URL: https://datasheets.maximintegrated.com/en/ds/DS3231.pdf (visited on 06/08/2021).

[146] VDE-AR-N 4100 Anwendungsregel: 2019-04. “Technische Regeln für den Anschluss von Kundenanlagen an das Niederspannungsnetz und deren Betrieb (TAR Niederspannung)”. Norm. Apr. 2019.

[147] *FT232H Single Channel High-Speed USB to Multipurpose UART/FIFO IC*. Future Technology Devices International Ltd. URL: https://ftdichip.com/wp-content/uploads/2020/07/DS%5C%5FFT232H.pdf (visited on 06/08/2021).

[148] Walt Kester. “MT-001: Taking the Mystery out of the Infamous Formula, ‘SNR=6.02N + 1.76dB’, and Why You Should Care”. In: *Analog Devices Tutorial*. 2005.

[149] *STPM32, STPM33, STPM3 - ASSP for metering applications with up to four independent 24-bit 2nd order sigma-delta ADCs, 4 MHz OSF and 2 embedded PGLNA*. STPM32, STPM33, STPM34. Rev. 5. STMicroelectronics. URL: https://www.st.com/resource/en/datasheet/stpm32.pdf (visited on 06/08/2021).

[150] *SCHRACK Power PCB Relay RT1 bistable*. TE Connectivity. URL: https://www.te.com/deu-de/product-8-1393239-4.html (visited on 06/08/2021).

[151] *2W Single Output Encapsulated Type IRM-02 series*. MEAN WELL. URL: https://www.meanwell.com/Upload/PDF/IRM-02/IRM-02-SPEC.PDF (visited on 06/08/2021).

[152] *ESP-IDF Programming Guide*. Espressif. URL: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/wifi.html%5C#esp32-wi-fi-throughput (visited on 06/08/2021).

[153] Thomas Picon, Mohamed Nait Meziane, Philippe Ravier, Guy Lamarque, Clarisse Novello, et al. “COOLL: Controlled On/Off Loads Library, a Public Dataset of High-Sampled Electrical Signals for Appliance Identification”. In: *arXiv preprint:1611.05803*. 2016.

[154] Peter Welch. “The Use of Fast Fourier Transformfor the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms”. In: *Transactions on Audio and Electroacoustics*. IEEE, 1967, 15.2.

[155] OASIS. “The MQTT 5.0 Standard - A Machine-to-Machine (M2M) "Internet of Things" Connectivity Protocol”. 2020. URL: `https://www.mqtt.org/` (visited on 06/08/2021).

[156] Francisco Tirado-Andrés and Alvaro Araujo. “Performance of clock sources and their influence on time synchronization in wireless sensor networks”. In: *International Journal of Distributed Sensor Networks*. SAGE Publications, 2019, 15.9.

[157] Network Time Foundation. “NTP: The Network Time Protocol”. 2020. URL: `http://www.ntp.org` (visited on 06/28/2021).

[158] WavPack. “Hybrid Lossless Audio Compression”. 2020. URL: `http://www.wavpack.com` (visited on 06/08/2021).

[159] The HDF Group. “The HDF5 library & File Format”. 2020. URL: `https://www.hdfgroup.org` (visited on 06/08/2021).

[160] Ulrich Greveler, Peter Glösekötterz, Benjamin Justusy, and Dennis Loehr. “Multimedia Content Identification Through Smart Meter Power Usage Profiles”. In: *Proceedings of the International Conference on Information and Knowledge Engineering*. IKE ’12. World Congress in Computer Science, 2012.

[161] Django Software Foundation. “DjangoThe web framework for perfectionists with deadlines”. 2020. URL: `https://www.djangoproject.com/` (visited on 06/08/2021).

[162] Non-Profit Organization Matroska. “The Matroska File Format”. 2020. URL: `https://www.matroska.org/` (visited on 06/08/2021).

[163] Paul M Fitts. “The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement”. In: *Journal of experimental psychology*. American Psychological Association, 1954, 47.6.

[164] Stuart K Card, Thomas P Moran, and Allen Newell. “The Keystroke-Level Model for User Performance Time with Interactive Systems”. In: *Communications of the ACM*. ACM, 1980, 23.7.

[165] David Kieras. “Using the Keystroke-Level Model to Estimate Execution Times”. In: *Journal of the University of Michigan*. 2001, 555.

[166] co2online. “Der Stromspiegel für Deutschland 2019”. 2020. URL: https://www.stromspiegel.de/stromverbrauch-verstehen/stromverbrauch-3-personen-haushalt/ (visited on 06/08/2021).

[167] Conrad Electronic AG. “Voltcraft Energy logger 4000”. 2020. URL: http://www.voltcraft.ch/index.html (visited on 06/08/2021).

[168] scikit-learn developers. “scikit-learn - Machine Learning in Python”. URL: http://scikit-learn.org (visited on 06/08/2021).

[169] Peter Hart. “The Condensed Nearest Neighbor Rule”. In: *Transactions on Information Theory*. IEEE, 1968, 14.3.

[170] GitHub. “XGBoost - eXtreme Gradient Boosting”. List of Machine Learning Challenge Winners based on XGBoost. 2020. URL: https://github.com/dmlc/xgboost/tree/master/demo#machine-learning-challenge-winning-solutions (visited on 06/08/2021).

[171] Vijay Pappu and Panos M Pardalos. “High-Dimensional Data Classification”. In: *Clusters, Orders, and Trees: Methods and Applications*. Springer, 2014.

[172] Rune Jacobsen, Nikolaj Torring, Brian Danielsen, Morten Hansen, and Erik Pedersen. “Towards an app platform for data concentrators”. In: *Proceedings of the PES Innovative Smart Grid Technologies Conference*. ISGT ’14. IEEE, 2014.