



CHALMERS
UNIVERSITY OF TECHNOLOGY

Detecting Appliances in Energy Traces

MIKAEL ALPING & ANTON BERGMAN

MASTER'S THESIS 2016

Detecting Appliances in Energy Traces

MIKAEL ALPING
ANTON BERGMAN



Department of Computer Science and Engineering
Division for Networks and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2016

Detecting Appliances in Energy Traces
MIKAEL ALPING, ANTON BERGMAN

Supervisor: Magnus Almgren, Networks and Systems Division, Department of Computer Science and Engineering, Chalmers University of Technology

Examiner: Olaf Landsiedel, Networks and Systems Division, Department of Computer Science and Engineering, Chalmers University of Technology

Master's Thesis 2016
Department of Computer Science and Engineering
Networks and Systems division
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 1000

Typeset in L^AT_EX
Gothenburg, Sweden 2016

Detecting Appliances in Energy Traces
MIKAEL ALPING, ANTON BERGMAN
Department of Computer Science and Engineering
Chalmers University of Technology

Abstract

The amount of data gathered in the world is increasing every day. More and more energy data is being gathered from households and smart energy meters. To extend the functionality of these energy traces, *Non-Intrusive Load Monitoring (NILM)* algorithms can be used. These algorithms use training data in the form of appliance-specific energy trace to label different sections of the aggregated energy trace with activity.

In this thesis, we investigate how to create a data set with the goal of using it to investigate NILM algorithms, and to build a platform for future student projects in the area of energy trace data sets. This platform contains suggestions on what methods to use for gathering data, how to store it, and how to analyse it.

Keywords: Big Data, Machine Learning, Cyber-Physical Systems, Smart Grid, Advanced Metering Infrastructures

Acknowledgements

We would like to thank our supervisor, Magnus Almgren, for guiding us through the project. We want to thank Omegapoint for allowing us access to their offices, and Martin Altenstedt for providing much needed technical expertise.

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 1 |
| 1.1 | Problem Definition and Goals | 2 |
| 1.2 | Limitations | 3 |
| 1.3 | Thesis Outline | 3 |
| 2 | Previous Work | 5 |
| 2.1 | Household Energy Data Sets | 5 |
| 2.1.1 | REDD | 5 |
| 2.1.2 | Smart* | 6 |
| 2.1.3 | BLUED | 7 |
| 2.1.4 | AMPDS | 7 |
| 2.1.5 | The ECO Data Set | 7 |
| 2.1.6 | Summary - Data Sets | 8 |
| 2.2 | Algorithms on Data Sets | 8 |
| 2.2.1 | Non-Intrusive Load Monitoring Algorithms | 8 |
| 2.2.2 | Algorithms Application on Data Sets (NILMTK) | 9 |
| 2.3 | Summary - Previous Work | 11 |
| 3 | Data Collection | 13 |
| 3.1 | Methodology | 13 |
| 3.2 | Data Set Design Goals | 13 |
| 3.2.1 | Energy Trace | 14 |
| 3.2.2 | Temperature | 14 |
| 3.2.3 | Humidity | 15 |
| 3.2.4 | Door Activity | 15 |
| 3.3 | System Design Goals | 15 |
| 3.3.1 | Price | 15 |
| 3.3.2 | Availability | 16 |
| 3.3.3 | Logging Speed/Resolution | 16 |
| 3.3.4 | Data Quality | 16 |
| 3.3.5 | Versatility | 16 |
| 3.3.6 | Prior Knowledge | 16 |
| 3.3.7 | Setup Time | 16 |

| | | |
|----------|---|-----------|
| 4 | System Architecture and Implementation | 19 |
| 4.1 | Data Gathering Systems | 19 |
| 4.1.1 | Raspberry Pi 2 Model B | 19 |
| 4.1.2 | Open Energy Monitor | 21 |
| 4.1.3 | CLM1000 Professional | 22 |
| 4.1.4 | Z-Wave System | 22 |
| 4.2 | Data Storage | 24 |
| 4.2.1 | Relational Database | 24 |
| 4.2.2 | Non-relational Database | 25 |
| 4.2.3 | Backup strategy | 25 |
| 4.3 | The Data Set | 25 |
| 4.3.1 | Household Data | 26 |
| 4.3.2 | Controlled Test Environment | 27 |
| 4.3.3 | Analysis Toolkit Data | 28 |
| 4.3.4 | Raw Data | 28 |
| 5 | Data Set Analysis | 31 |
| 5.1 | Visual Analysis | 31 |
| 5.2 | Data Set Comparisons | 31 |
| 5.3 | Analysis Toolkit NILMTK | 32 |
| 6 | Results | 33 |
| 6.1 | Data Gathering | 33 |
| 6.1.1 | Grades | 33 |
| 6.1.2 | Z-Wave System | 34 |
| 6.1.3 | Open Energy Monitor | 35 |
| 6.1.4 | CLM1000 Professional | 36 |
| 6.2 | Analysis | 37 |
| 6.2.1 | Visual Analysis | 37 |
| 6.2.2 | Data Set Comparisons | 40 |
| 6.2.3 | Analysis Through NILMTK | 40 |
| 7 | Ethics | 41 |
| 7.1 | Surveillance | 41 |
| 7.2 | Sustainability | 41 |
| 8 | Lessons Learnt | 43 |
| 8.1 | Hardware | 43 |
| 8.2 | The Data Set | 43 |
| 8.3 | Software | 43 |
| 9 | Extensions | 45 |
| 9.1 | The Data Set | 45 |
| 9.2 | Refrigerator Model | 45 |
| 9.3 | Analysis | 45 |
| 9.4 | Alternative use of Energy Traces | 46 |

| | | |
|-----------|---|-----------|
| 10 | Conclusion | 47 |
| | Bibliography | 49 |
| A | Appendix 1 | I |
| A.1 | CLM1000 Professional Automatisation Script | I |
| A.2 | CLM1000 Professional Upload Script | II |
| A.3 | Z-Wave Upload Script | III |
| A.4 | Raspberry Pi Hardware Spec | IV |
| A.5 | Z-Wave Hardware Spec | V |
| A.6 | Open Energy Monitor Hardware Spec | V |
| A.7 | CLM1000 Professional Hardware Specification | V |

1

Introduction

In today's society, immense amounts of data is collected every day. This data can be produced from a variety of sources. One such source is from objects that people use in their everyday life. A popular description of this phenomenon is the *Internet of Things*, where physical objects are embedded with electronics to make the object able to store data and communicate by exchanging this data amongst other objects and with the rest of the connected world [8].

Applications of the Internet of Things include:

- Extensive information about devices, both used in industrial settings and for personal use, to predict whether the device will break down ahead of time.
- Automated operation of devices, such as turning off the lights in a residential home if no one is currently there.

These applications can help to benefit both industrial applications as well as regular customers. The applications can also contribute to reduce the energy consumption of devices in general, as more control and more insight of the device is given to the users.

In light of the Internet of Things, a vast amount of energy trace data from households is today being gathered by using smart energy meters. [22] As of today, this kind of data is mainly being sent from households for the purpose of electrical companies billing their customers. However, if this aggregated household energy trace could be collected and separated into its parts of appliance-level energy traces, more information could be learnt about the household consumption with the goal of either reducing the total consumption, or moving it to times when there is more energy available in the system. Using the aggregated household energy trace in such a way would thus be much cheaper than collecting the same information by monitoring each appliance individually.

To distinguish the contribution of a device from the aggregated household energy trace, so called *Non-Intrusive Load Monitoring*, or *Energy Trace Disaggregation*, algorithms can be used. Most such algorithms need training data, in the form of plug-level data from household appliances, to eventually be able to distinguish devices of that type from generic aggregated household energy traces. [13]

Batra et al. [5] present three motivations for disaggregating energy traces of households:

- Household occupants can be notified of how much energy each appliance in their homes consumes, and will thus be motivated to take steps to reduce their energy consumption.
- Individual appliance feedback can be more accurately provided, emphasising appliance-specific advice, such as potential financial gain when replacing an old appliance with a more efficient one.
- Non-Intrusive Load Monitoring algorithms are able to determine the time of use for each appliance, and thus can recommend the optimal time of usage when electricity is cheaper or has a lower carbon footprint.

1.1 Problem Definition and Goals

How can a large set of energy consumption data be gathered in a limited time and at a reasonable financial cost for training and validation of Non-Intrusive Load Monitoring algorithms? What requirements needs to be fulfilled for such a data set to work efficiently with existing analysis methods?

The goals of this master thesis are to:

1. Investigate how to create a system for collection of energy consumption data from appliances found in typical residential apartment and houses. We focused our work on refrigerators and freezers, but the methodology can be applied to other appliances as well.
2. Create a platform for data gathering and data analysis that is easy to deploy and can be used as a basis in the future to collect more data in other projects.
3. Use the above system to create a rich energy data set to be used for energy consumption trace analysis.
4. Evaluate the quality of the collected data set, by using a selection of existing Non-Intrusive Load Monitoring analysis algorithms on the data set.

Taken together, points 1-4 will contribute to reducing the startup time for future data collection efforts, and help to provide a base of knowledge for future energy trace data and Non-Intrusive Load Monitoring projects.

Many existing solutions to the problem of gathering energy trace data from households require extensive setup and are often very expensive. However, a simpler setup can also influence the quality of data, which means that we investigate the trade-off of the setup with the data quality (step 3). In our work, we have considered the full system (step 2), investigating and building code for data gathering, data storage, and finally algorithms for analysis and data extraction.

1.2 Limitations

The data set is restricted to data from refrigerators. However, the methodology would be similar for other appliances as well. Refrigerators in particular were chosen for a couple of reasons. First, it is an appliance that is likely to be present in almost all residential homes. Furthermore, it is also an appliance that has a large energy footprint when compared to other appliances in the home. Finally, it is also an appliance where several types of data can be collected and used to model its behaviour such as temperatures and door openings.

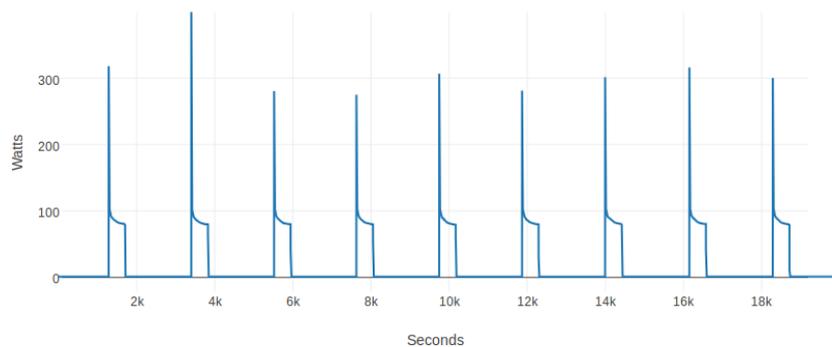


Figure 1.1: *An example of the energy trace of an empty refrigerator*

Due to the somewhat short time frame of this Master thesis in comparison to previous work in this field, the work is focused on building a system to collect high quality data and the evaluation of the final data sets using our proposed methodology. We do not suggest new algorithms for data analysis, but survey existing literature.

1.3 Thesis Outline

Chapter 1 Contains an introduction to the thesis, project motivations and project goals.

Chapter 2 Describes previous work in the area of household data set gathering and analysis.

Chapter 3 Contains theory behind the gathering of data, both from households in general and from refrigerators in specific.

Chapter 3 Describes the different tools and methods we used to gather our data, and describes the different data gathering experiments that was conducted.

Chapter 5 Describes the methods used to evaluate and analyse the data set gathered.

Chapter 6 Describes the results achieved from data gathering and data analysis.

Chapter 7 Presents ethical parts of the thesis, such as environmental and surveillance-related questions.

Chapter 8 Consists of the conclusions drawn from both data gathering and data analysis.

Chapter 9 Describes some lessons learnt while conducting the project.

Chapter 10 Presents possible extensions to the thesis and the data gathering platform.

2

Previous Work

There has been some previous work on the topic of creating data sets of household energy consumption data and extracting relevant information from them using analysis algorithms. In this chapter, we describe previous work that is relevant and how we intend to use previously conducted research in this thesis.

2.1 Household Energy Data Sets

We here present a number of previously conducted household data gathering projects and the characteristics of their data sets. This material is used to first guide our own methodology for data collection, and second as quality control where we can compare the data set collected in this thesis with other existing data sets.¹

2.1.1 REDD

Kolter and Johnson [17] present the Reference Energy Disaggregation Data set (REDD). At the time of writing their paper (June 15th, 2011), they had gathered in total 119 days worth of data from 10 households. For each household, three different levels of energy data are gathered: the total aggregated household energy trace, different aggregated sub-circuits in the household, and plug-level energy data. The aggregated trace is measured at 15kHz, the circuits at 0.5Hz, and plug-level data at 1Hz. The high level aggregate measurements are due to very sophisticated high-end measuring equipment.

Kolter and Johnson also provide some insight into the design choices made when suiting their data set best to available disaggregation algorithms. Their choice of frequency for gathering the aggregated household trace is justified with that higher frequency data can always be sub-sampled to lower frequency data if wanted. On the other end of the spectrum, the data must always be feasible to store in a reasonable way.

The same argument is used for whether both real and reactive power should be gathered. They collect the AC waveform as both real and reactive power can be computed from it.

¹Due to one of the goals of this thesis being to gather a light-weight data set with certain limitations applied, a comparison that can be made against most other data sets is that our data set will be of smaller quantity, but this is to be expected. Other differences are described.

No external features, such as time of day or weather information, is gathered except for UTC time stamps and city location, as most other relevant data can be deduced from this.

As disaggregation algorithms often require *supervised* training data [6], REDD includes as much supervised information as possible. Each circuit of the household is gathered and labelled to help provide more specific information about the household.

Finally, to generalise their data as much as possible, REDD includes data gathered from many households on many different types of devices. A wanted feature of disaggregation is to have as much general data as possible. This to allow for the algorithms to recognise a certain type of device, rather than a very specific device, and would thus not require as much training data.

Even though a lot of detailed data is gathered from many different homes, REDD does not contain any kind of controlled test environments for gathering data but instead only natural household data.

2.1.2 Smart*

Barker et al. [3] present the open and publicly available² *Smart** data set, which is divided in two parts.

The first part is gathered from three households. The aggregated trace is gathered at a frequency of 1Hz. More in-depth information about the households such as plug loads, circuits, wall switches and motion sensors, is gathered at different higher, but unspecified frequencies. The second data set is much larger, but also of lower quality, gathered from 400 households with only the aggregated trace gathered at 1Hz. The *Smart** data set was gathered over a period of three years.

The focus of *Smart** is depth over breadth, which means aiming for more detailed data rather than data from many different sources. They have measured, in addition to power consumption, variables such as temperature, door events, outdoor weather data, etc.

Barker et al. describe REDD as an inspiration for their project, but also state that *Smart** differs from REDD in three important ways: *heterogeneity*, *scalability* and *redundancy*. For heterogeneity, they gather not only energy data from the mains panel, circuits and plugs, but also include data from sensors related to the devices, such as weather data, motion data, door openings, thermostat switches, etc.

For scalability, one goal of the *Smart** data set is to collect high-resolution data at a large scale. Barker et al. believe that collecting data from as many loads in a household as possible will enable new types of research, and thus data is gathered even from smaller devices with a load under 50 watts.

For redundancy, the same data is gathered at different "levels" of the electrical tree; the entire home, each circuit, and each plug, which in turn reveals important information about the relative accuracy of the sensors.

²<http://traces.cs.umass.edu/index.php/Smart/Smart>

2.1.3 BLUED

Anderson et al. [1] present BLUED (Building-Level Fully-Labeled Data Set for Electricity Disaggregation). BLUED is made up of voltage and current data from a household during one week. The data is gathered at a relatively high frequency, 12kHz. State transitions of the appliances found in the data set are labeled and time-stamped to further ease the work of applying analysing algorithms to it. It is also publicly available on their website.³

Beside presenting the tools for gathering the data, Anderson et al. also present some reflections on the challenges they met when gathering their data set.

2.1.4 AMPDS

AMPDS (The Almanac of Minutely Power Data Set), as presented by Makonin et al. [20], consists of one year's worth of data sampled from 21 sub-meters at 17mHz (1 minute intervals). Makonin et al. talk about some key problems that need to be solved in a good way to determine the success of load disaggregation systems, and how their data set focuses on one of them: *dynamic and changing usage*.

Dynamic and changing usage means that the number of appliances, and the way that each appliance is used can change over time.

They further emphasise how this is a very accurate representation of how appliance usage works in the real world, and that it is a feature of data sets that should be able to be handled by disaggregation algorithms in a good fashion.

For each sub-meter, they measure a number of different parameters, such as Current, Voltage, Real/Reactive Power/Energy, etc.

2.1.5 The ECO Data Set

Beckel et al. [6] and Kleminger et al. [15] both describe a data set named the *Eco Data Set*, which contains detailed aggregate energy traces and household occupancy information from six different households.

The data in the Eco Data Set was collected over a period of 8 months, at a granularity of 1Hz, and contains aggregate electricity data with both real and reactive power, plug-level measurements from most common household appliances (such as refrigerators, dryers, kitchen appliances, electrical appliances), and occupancy information of the monitored households. The data set is publicly available for anyone to download of their website.⁴

Becket et al. compare their data set to REDD, the Smart* data set, BLUED, AMPDS, and a few others, and claim that the ECO data set extend these data sets in four aspects: the duration of the data gathering, the quality of aggregate energy data (both real and reactive power), and the quality of plug level data, and that they have the only data set that also includes occupancy information from households.

³<http://nilm.cmubi.org>

⁴<https://www.vs.inf.ethz.ch/res/show.html?what=eco-data>

Table 2.1: Summary of the data sets. Frequency is presented as household aggregate/circuit level/plug level.

| Name | Collected by | Frequency (Hz) | Length | Households |
|--------|------------------------|----------------|----------|------------|
| REDD | Kolter and Johnson[17] | 15k / 0.5 / 1 | 119 days | 10 |
| Smart* | Barker et al. [3] | 1 / 1 / 0.33 | 3 years | 400+ |
| BLUED | Anderson et al. [1] | 12000 / - / - | 8 days | 1 |
| AMPds | Makonin et al. [20] | - / - / 0.02 | 1 year | 1 |
| ECO | Beckel et al. [6] | 1 / - / 1 | 240 days | 6 |

2.1.6 Summary - Data Sets

In summary, many of the data sets look very much alike with small differences in, among other things, duration, frequency, and number/type of appliances/households. The three largest data sets are REDD, Smart*, and the ECO Data set, with 119 days, 3 years, and 8 months worth of data respectively⁵.

The REDD data set is gathered at different frequencies, all equal to or above 0.5Hz. ECO is gathered at 1Hz. Smart* gathers household electricity usage and circuit data at 1Hz, and gathers plug level data every other second.

BLUED is a slightly smaller data set at only a week’s worth of data, but at a really high frequency, 12kHz. AMPds is also a year of gathered data, but at a much lower frequency at 1 sample per minute. A summary of all the data sets can be found in Table 2.1.

2.2 Algorithms on Data Sets

In this section, a number of previously researched analysis methods and algorithms on large household energy trace data sets are presented. For each method and algorithm, their context, how they are implemented/used, and what results they provide are discussed.

2.2.1 Non-Intrusive Load Monitoring Algorithms

The first implementation of non-intrusive load monitoring algorithms was presented by Hart [13]. The algorithm presented by Hart aims to match changes, in the form of sharp edges, in the aggregate energy trace with entries in a signature database to learn shifts in the states of different appliances. The signature database is manually filled with different characteristics of appliances and no learning algorithms are used.

Since then, different variations of NILM algorithms have been presented, and will be presented below. Many of the data collection papers described in Section 2.1 also apply a number of these algorithms on their data. For example, Becket et al. [6] implement and present four different NILM algorithms: Parson’s Algorithm, Baranski’s Algorithm, Weiss’ Algorithm and Kolter’s Algorithm.

⁵For some of these data sets, the collection is ongoing and might be larger now.

Table 2.2: Algorithms Summary

| Algorithm | Supervised/Unsupervised | Frequency (Hz) |
|----------------------|-------------------------|----------------|
| Parson’s Algorithm | Semi-Supervised | 0.02 |
| Baranski’s Algorithm | Unsupervised | 1 |
| Weiss’ Algorithm | Supervised | 1 |
| Kolter’s Algorithm | Unsupervised | 1 |

Parson’s Algorithm [21] is built using *Hidden Markov Models* and the *Viterbi Algorithm* [23] to calculate for each appliance in a given trace the most likely sequence of operating states. Given this information, the algorithm tries to estimate the consumption of the appliance and subsequently separate it from the total energy trace. Parson’s Algorithm is *semi-supervised*, as it instead of using specific appliance-level energy data uses some generic characteristics of appliances. Knowledge is thus gathered from averages and expected values instead of known previously gathered values.

Baranski’s Algorithm [2] finds recurring patterns of the total energy trace and pairs them together with their respective appliance. The total energy trace is looked at, and changes over certain thresholds for each appliance is clustered. From this clustered data, a state machine is built and the most probable state sequence for each appliance is calculated. This algorithm is unsupervised and works without knowing in beforehand what appliances exist within the trace, but each cluster created by the algorithm must be assigned manually to an appliance.

Weiss’ Algorithm [24] recognises changes in the energy trace and assigns these changes to a specific appliance using a signature database. This is based on the same approach as presented by Hart, as it clusters events while using training data, and then assigns events to the appliance with the best matching cluster during real operation.

Kolter’s Algorithm [16] also uses Hidden Markov Models, but it is unsupervised and does not require any previous information about the appliances in the trace. The algorithm estimates the number of appliances and their energy trace given the aggregated energy trace, and then creates a HMM from this information.

Table 2.2 presents a summary of the different algorithms presented by Becket et al. [6]. The Data Logging Frequency in the table is the frequency of data which Becket et al. used the algorithms on. It is somewhat unclear as to why they decided to use Parson’s Algorithm on data with much lower granularity, but they do explain in their results that better data granularity makes the algorithms perform better, and Parson’s algorithm performed the worst of the algorithms they used.

2.2.2 Algorithms Application on Data Sets (NILMTK)

Batra et al. [5] present the open-source *Non-Intrusive Load Monitoring Toolkit (NILMTK)*, which allows for easy result analysis of non-intrusive load monitoring algorithms on a diverse range of data sets.

The data sets available for testing with NILMTK are: REDD [17], BLUED [1],

Smart* [3], the Pecan Street Data Set [14], the Household Electricity Survey Data set (HES) [25], the Indian data for Ambient Water and Electricity Sensing (*iAWE*) [4], and the Almanac for Minutely Power data set (AMPDS) [20].

To analyse these data sets, *NILMTK* provides a pipeline with four stages:

1. First, the data is converted to a common data format, *NILMTK-DF*. This data format is loosely based on the *REDD* data set format. Batra et al. also provide parsers from six of the available data sets to *NILMTK-DF*.
2. Second, the data can be evaluated using a number of diagnostics and statistical methods. These methods can detect information such as empty gaps in the energy trace, the dropout rate (recorded samples divided by expected samples) of the energy traces and the uptime of the system.
3. The data is then preprocessed to try to mitigate problems. If a data set contains data gathered at different frequencies, *NILMTK* can down sample the data to some common frequency. Some countries use different standard voltages, so *NILMTK* can normalise the voltage of data sets.
4. Finally, the top k most energy consuming appliances can be found instead of trying to separate the aggregated trace into all its sub-parts.

The two disaggregation algorithms implemented by *NILMTK* are combinatorial optimisation (CO) and Factorial Hidden Markov Models (FHMM). The combinatorial optimisation algorithm presented by Batra et al. is the same as proposed by Hart [13]. CO aims to find the optimal combination of appliance states to minimise the difference between the observed aggregate energy trace and the sum of all predicted appliance energy traces.

Hart explains this to be the same as the NP-complete *weighted set* problem [10], and thus an approximation algorithm should be used for any realistic problem size. Batra et al. explain the complexity of disaggregation for T time slices to be $O(TK^N)$, where K is the number of appliance states, and N is the number of appliances.

The Factorial Hidden Markov Model algorithm models the expected energy trace of each appliance as the observed value of a Hidden Markov Model (HMM), with the hidden part of the *HMM* being the appliance's states. To disaggregate an energy trace, the power consumption of a number of appliances must be jointly decoded. This makes *FHMM* well suited for the task [11]. The complexity for disaggregating an energy trace using FHMMs is $O(TK^{2N})$, which means that it scales worse than CO.

As NILM algorithms have many areas of application, *NILMTK* uses a set of metrics to judge the accuracy of the algorithms after applying them to data sets. The accuracy metrics are:

Error in energy assigned: The error between the total energy assigned to the appliance by the algorithm and the actual energy.

Fraction of energy assigned correctly: The fraction of the total energy trace of the household assigned per appliance, and the actual fraction.

Normalised error: The sum of differences between assigned and actual power per appliance, normalised by the appliance's total consumption.

Root mean square error: RMS error between assigned power and actual power of each appliance.

Confusion Matrix: Number of time slices where an appliance were either correctly classified or confused with every other state.

NILMTK also presents the number and fraction of time slices that were predicted wrongly/correctly to be on/off. Finally, **Hamming Loss** is presented. This is the total information lost when appliances are incorrectly classified.

2.3 Summary - Previous Work

This chapter surveyed existing data sets and a set of NILM algorithms previously suggested in literature. It also explained the function of the open-source software suite NILMTK to analyse data. As can be seen, there exists software to test algorithms on data sets without having much in-depth knowledge of energy trace disaggregation algorithms.

We will return to the papers and techniques researched in this chapter when we discuss our data set collection methodology in Chapter 3, and when assessing the quality of our data in Chapter 6.

3

Data Collection

In this chapter, different methods and tools for gathering energy trace data from household appliances with the goal of building a larger coherent data set are analysed, to justify the choices we made for our system in Chapter 4.

3.1 Methodology

As explained in Chapter 2, several household energy trace data sets already exist, but these have often been collected with different goals in mind than those of this thesis. To choose which system should be used to gather the data set for this thesis, three properties were considered:

- **Cost:** The system should be as cheap as possible, while still providing the rest of the necessary properties.
- **Usability:** The system should be easy to use by anyone who could want to extend the data set.
- **Extensibility:** The system chosen to gather our data set should be able to connect as many sensors as possible to the same platform, as to avoid hardware redundancy.

We structure the chapter along the data collection for refrigerators. The methodology, even though applied on a certain kind of appliance in a certain way, is general and can be adapted to other types of appliances as well.

3.2 Data Set Design Goals

An aggregated energy trace from a household contains the total energy consumption of all the appliances in the household. To separate individual appliances in the trace, the algorithms used need to recognise the pattern of the individual appliance's energy traces.

In this study we focus on gathering energy consumption data from refrigerators, because they are very common and are present in most residential homes. They also have a rather large energy consumption footprint, compared to other household appliances. Several other types of data can also be gathered and can be used to

model the refrigerators behaviour. Refrigerators also have a very characteristic energy trace, see Figure 1.1. To be able to accurately model the appliances, and especially the refrigerator we need to understand how it works and use a model to describe under what circumstances it turns on.

The following section describes a simple refrigerator model used in this thesis. It includes the most important aspects that affects the energy trace of a refrigerator.

3.2.1 Energy Trace

All electronic devices have an energy consumption that can be measured in watts. The aggregated energy usage is called the energy trace of such a device. The energy trace can be represented using a graph that shows energy usage in watts on the y-axis and time on the x-axis.

The energy trace of a refrigerator follows a very specific pattern. While the temperature inside the refrigerator is below a certain threshold, the energy trace is flat at around 1W. When the temperature raises above this threshold, the refrigerator turns on and a very characteristic spike is formed in the energy trace, as can be seen in A in Figure 3.1. This spike is observed to be anywhere between 200-850 watt, depending on the refrigerator, and it then stabilises around 80-100 watt until the refrigerating unit turns off. The refrigerator unit stays turned on until the temperature drops below a lower threshold, shown as B in Figure 3.1. Finally, the refrigerator as a whole resumes a static energy trace at around 1W, as can be seen in C of Figure 3.1.

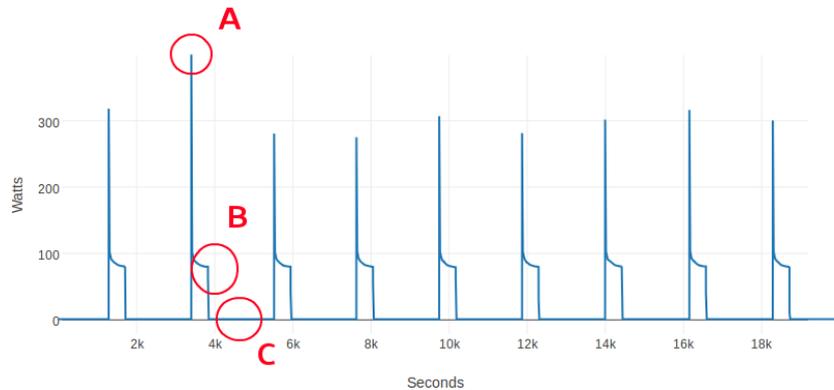


Figure 3.1: *A characteristic refrigerator energy trace, marking out the different noticeable state shifts.*

3.2.2 Temperature

The temperature of the refrigerator is highly correlated to the energy trace. When the temperature raises above a certain threshold, the refrigerating unit turns on and starts to lower the temperature. As described by Gupta et al. [12], the temperature

in the entire refrigerator is not always the same. Often, a lower compartment for vegetables and such is slightly warmer than the rest of the refrigerator. Thus, for an accurate model of the temperature(-s) within the refrigerator, one need several sensors placed at different locations.

Such thermometers also need to provide data at a frequency that catches any drop in temperature as it happens. Ideally the temperature should be measured at a similar rate to the energy consumption.¹

3.2.3 Humidity

The humidity in a fridge also impacts the energy consumption. The evaporated water acts as a heat transfer agent and the energy consumption can be directly connected to the humidity in a fridge [18]. However, humidity is difficult to measure accurately with off the shelf sensors.

3.2.4 Door Activity

Opening and closing the door of the refrigerator causes cold air to exit the refrigerator and warm air to enter, thus causing the temperature inside to rise more quickly. Consequently, the refrigerating plant will turn on more often, which in turn causes the pattern in the energy trace to occur more often.

3.3 System Design Goals

To create a system that can be used by students, hobby enthusiast and other groups with a more restrictive knowledge and time pool, we had a set of system design goals that we used when we created the system used in this thesis. All systems that were surveyed were also evaluated according to these system goals. This evaluation helps fulfil point 2 of the problem definition from Section 1.1.

Table 3.1 presents a summary of the design goals of the data collection tool with a brief description of how the design goal should be considered.

3.3.1 Price

In this project we decided to keep low cost as an overarching design goal. This will enable home makers, students, and others to build their own system without a large investment into sensor technology. This is an important part due to the often restricted budget of such projects. A low price allow for wider adoption which in turn will lead to a richer data set.

¹Note that this have not been done in this study, due to the fact that the Z-Wave system does not provide an accurate enough temperature sensor, see Section 4.1.4.

3.3.2 Availability

Another important part for projects with a large impact/user base, is how easy it is to obtain the system. When time is a factor, a group cannot wait several weeks for a system to arrive. During the design we strove to use systems that could easily be found in common electronic stores. We also strove to find sensors that worked out of the box and that did not require further assembly and deeper technological knowledge.

3.3.3 Logging Speed/Resolution

The logging speed of a system is in direct correlation with the analysis. Slow logging speed and low resolution of data may cause important patterns in the energy trace to be lost. NILM algorithms also have different requirements on resolution [6].

The algorithms are summarised in Table 2.2. Often a higher logging speed is more desired, since the data can always be down-sampled to fit other purposes.

3.3.4 Data Quality

With data quality we refer to how precise the collected data is. It needs to be precise enough to cover the phenomena aimed to measure. For example, the energy meters need to be able to record the energy precise enough to catch the refrigerators' characteristic pattern and work for several weeks without crashing. Data quality also means that the data should be "mostly continuous", which means that wireless devices need to be able to transfer data without major package loss.

3.3.5 Versatility

To be able to create a rich data set and create a complete model of the appliance (in this case the refrigerator) we need to create a system that can handle a wide range of sensors. These sensors should preferably come from the same system so that several systems do not need to be connected or run in parallel.

3.3.6 Prior Knowledge

Different systems have different levels of prior knowledge requirements. For example, some systems require more electrical systems knowledge and some require more of a programming background. For our purposes in this thesis, we strive for systems that do not require any particular expertise in an engineering topic, but instead should work as "plug and play".

3.3.7 Setup Time

The Setup Time is the time from the time of starting the work until the point that data is being gathered. We aim for the Setup Time to be as low as possible. As

Table 3.1: Design Goal Summary

| Design Goal | Target Level |
|--------------------------|---|
| Price | As low as possible without compromising data quality. |
| Availability | Available to anyone from general stores. |
| Logging Speed/Resolution | To fit analytics tools and algorithms. |
| Data Quality | Enough to measure specific target devices. |
| Versatility | More sensors should be able to be added. |
| Prior Knowledge | No specific knowledge should be needed. |
| Setup Time | As low as possible. Off-the-shelf. |

presented by the goals in Section 1.1 a result of this master thesis will hopefully be to reduce the setup time overhead for future efforts.

4

System Architecture and Implementation

In this chapter, the system used to collect data is presented. This includes the different hardware and software used as well as motivation as to why these choices were made. Two different database technologies are discussed, and are compared in terms of structure and performance. Finally, the data set is presented in its entirety, and the contents and more general structure of the data set is discussed.

4.1 Data Gathering Systems

The data gathering systems consist of four parts: the appliance, the sensors, the relay device and the database. The system is designed to work with any appliance that uses a regular wall plug; in this project the appliance is a refrigerator.

Connected to the refrigerator is one or more sensors. We in this case use sensors to measure plug level energy consumption data from the refrigerator as well as temperature and door openings. The sensors talk to a relay device, which stores the data locally and sends the aggregated data every five minutes to the database.

The energy data was collected using two different systems, however three systems was reviewed. The first was an off the shelf energy meter called CLM1000 Professional (Section 4.1.3), which seemed easy to use, but had no extensions. The second was the Open Energy Monitor which had the potential to be extendable and low level. The third system was based on sensors that use the Z-Wave protocol (Section 4.1.4), which was somewhere in between the two other systems. The systems are split into three parts: sensors, a relay device and a database. (Figure 4.1)

The sensors collect the data from an appliance and send it to the relay device. The data is continuously sent to the relay device, which temporarily stores it, before sending the aggregated data to the database. ¹

4.1.1 Raspberry Pi 2 Model B

A Raspberry Pi 2 Model B is a small general purpose computer. We use it to communicate with the sensors and to temporary log data before sending it to a database. The data is sent every five minutes to the database. The Raspberry Pi

¹Open Energy Monitor (Section 4.1.2), was discarded due to a number of problems.

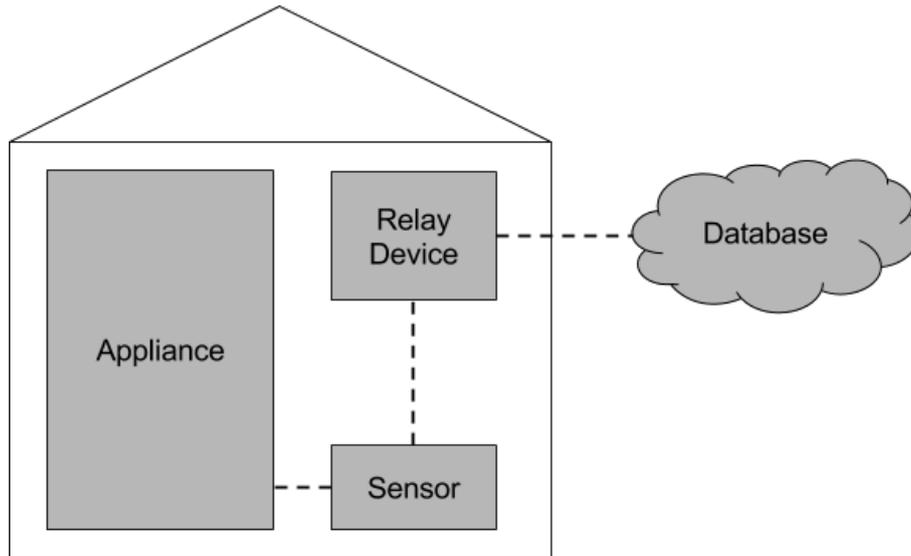


Figure 4.1: *System Overview*

runs one of two Python scripts (depending on system) developed to automate the data flow between sensors and the database. The source code can be found in the appendix (Appendix A.2 and A.3). We choose the Raspberry Pi because it is a small device that can easily be stored in a household, and because it is easy to obtain.

The Raspberry runs either the Raspbian operating system or a special made operating system from Z-Wave if Z-Wave sensors are used. Technical specs for the Raspberry Pi model used is found in Appendix A.4.



Figure 4.2: *The Raspberry Pi with the RaZberry chip*

4.1.2 Open Energy Monitor

Open Energy Monitor (OEM) shown in Figure 4.3 is an open source energy monitoring tool. It uses a Raspberry Pi, with a RFM12Pi chip attached, to communicate via radio to connected sensors. To measure energy consumption, *OEM* uses current transformers (Figure 4.4), which are clamped around the current carrying cable. The CT-sensors are connected to the EmonTx transmitter, which sends the data to the Raspberry Pi. The EmonTx can both be run with battery or with a power cord.

The system uses a special tailored Raspbian-based operating system, and comes with all the software necessary for data collection. The Raspberry Pi runs a server which logs the data locally and can be accessed via a web-application. Similar to the Z-Wave system, described below (Section 4.1.4), several other sensors can be added to the system, such as temperature and humidity sensors.



Figure 4.3: *The Open Energy Monitor*

Note: This system was not part of the final solution due to problems recording data from home appliances. The current transformer requires single current carrying cables, and most home appliances have multi-core cables. However the system was carefully reviewed and has a lot of potential. This is discussed more in data gathering results (Section 6.1) and conclusion (Chapter 10).



Figure 4.4: *The Open Energy Monitor CT-sensor*

Table 4.1: The data stored by the CLM1000 Professional

| CLM1000 Data Block |
|---|
| Timestamp |
| Active Power [W] |
| Apparent Power [VA] |
| Reactive Power [Var] |
| Tension [V] |
| Current [A] |
| Power Factor |
| Active Energy [kWh] |
| Apparent Energy [kVAh] |
| Reactive Energy [kvarh] |
| Identification of Load [resistor, inductivity, capacitor] |

4.1.3 CLM1000 Professional

The CLM1000 Professional is an off the shelf energy meter that utilises a plug between the appliance cord and the wall socket to log plug-level data. The CLM1000 can log data for up to 24 hours locally or send data continuously via USB, both at a rate of 1 Hz. It can be used with free software from the distribution company, however this is not compatible with the Raspberry Pi. Instead we used a Python script that reads directly from the USB-port.

Each energy reading from the CLM1000 contains several different types of data. Of all the surveyed energy meters it has the most accurate and consistent energy readings. The data is stored as a data block which is presented in Table 4.1. The data is stored as a text file on the Raspberry Pi.



Figure 4.5: *The CLM1000 Professional*

4.1.4 Z-Wave System

Z-Wave is an international standard, developed for wireless smart home communication. The system used in this project uses a Raspberry Pi with a RaZberry

communication chip, and several Z-Wave sensors. The Raspberry Pi runs an operating system made for Z-Wave, with a local server and Home Center application. The Home Center application is a web-based graphical user interface where sensors can be managed. The data gathered by the sensors are stored locally in JSON format files. The JSON files are read and parsed with Python scripts and the data is sent to an external database. We choose this system because of the many available sensors and because it is available in most electronic stores. It also seemed to come in between Open Energy Monitor and CLM1000 Professional in terms of complexity of usage and extensibility.

The Z-Wave system is a multipurpose system and uses several different sensors for gathering energy traces, temperature and door openings. This section contains information on how these sensors are utilised, how they work and problems that may arise during usage. For technical specification of all sensors, see appendix (A.7).

Energy Meter Aeon Labs ZW075-C16

The energy meter from Aeon Labs is a small device that is plugged into the wall socket to. The appliance that is being monitored is then plugged into the energy meter. The energy meter uses the Z-Wave protocol described above. This was the first wall plug used in the project, but was later replaced due to its incapability to measure energy consumption faster than once every three seconds (1/3 Hz).

Fibaro System Wall Plug

The Fibaro Wall Plug is also an extension of the wall socket, to which you connect the appliance to be measured. The wall plug uses the Z-Wave protocol and has the capability to measure and communicate the energy consumption once every second (1 Hz). This is one of the sensors included in the final configuration of our energy collection system.

Fibaro Door & Window Sensor

The Fibaro Door and Window sensor is a two piece device that is placed on a door or a window. When the two parts are separated it sends a signal "on" using the Z-Wave protocol and when put together it sends a signal "off".

DS18B20 Digital Thermometer

The DS18B20 is a small digital thermometer that connects to input ports on the Fibaro Door & Window Sensor. It then uses the Fibaro Door & Window sensor to send data to the Raspberry Pi. The DS18B20 is waterproof and can be sustained in harsh condition, making it perfect for use in a fridge. However it only sends data when temperatures are changing every 15 minutes. This makes it hard to get precise data and correlate it with the refrigerating unit.

NorthQ Power Reader

The NorthQ Power Reader is a power reader that can be used to monitor the total energy consumption of a household. It reads the pulses from the optical port that can be found on most home smart-meters or fuse boxes. The NorthQ power reader measures in kWh. This makes it hard to get a good aggregated energy consumption of a household in terms of resolution. We also had problems getting good power readings from the students apartments we tried this device in. Due to the low energy consumption of such a household, the energy meter rarely gave any readings at all.



Figure 4.6: *The Z-Wave sensors, Top: Fibaro Door/Window Sensor and DS18B20 Digital Thermometer, bottom left: NorthQ Power Reader, bottom middle: Fibaro System Wall Plug, bottom right: Energy Meter Aeon Labs ZW075-C16.*

4.2 Data Storage

When collecting large amount of data it is important to structure and store the data in an efficient and comprehensible way. To review different alternatives of data storage we decided to run two types of databases; a relational database and a non-relational database. The main difference between the two is that the first has a predefined schema, with relations between the data, and the latter has a dynamic schema with unstructured data [7].

4.2.1 Relational Database

Relational databases are well known and have been the standard for a long time. The data in a relational database is often easy to survey and handle. We chose to use a MySQL database, which is an open source relational database. The main reason we chose MySQL, was because we had some previous knowledge working with it. Other choices could also have been used such as PostgreSQL, and Oracle.

The MySQL database stores the data from the controlled test environment,

described in Section 4.3.2. Every test has its own table, named after what kind of test was performed.

4.2.2 Non-relational Database

Non-relational databases have gained popularity in the last few years. This is due to the fact that they were developed to handle big data very well. They also support horizontal scaling which makes them a lot faster than relational databases [7]. We chose to use MongoDB, which is a open source non-relational database.

The non-relational database uses MongoDB. The MongoDB database stores the data from households everyday use, as described further in Section 4.3.1. The database contains two collections. One named "*Location*" which contains information about where the appliance is located, the name of the tool that was used to record the data, and what kind of tool was used. It also contains appliance type and appliance model. The second collection contains all the sensor-data. The looks of the data can vary, but all contain energy consumption in watts, location ID, and device ID.

4.2.3 Backup strategy

An important part of storing large amounts of data is to have a good backup strategy. We choose to use a 3-2-1 backup plan. This means we store 3 different copies, on 2 different medias, and 1 backup stored off-site. This is done automatically, using bash-scripts run in a crontab.²

4.3 The Data Set

The data set is divided into four parts: household data, controlled test environment data, NILMTK-data and raw data. The main part of the data set is household energy data as it was collected from many households over a longer time. The household data is realistic data collected to monitor the every day use of fridges. The controlled test environment (*CTE*) data however is collected during a period of between 8 hours and a week, depending on the test. The CTE data is meant to test the behaviour of a fridge. The NILMTK-data is data tailored to work with the NILM tool kit, and is the same data gathered from the households but formatted differently. The raw data is all the data collected during the project, however it is not tailored and modified for any specific purpose.

The data set currently consists of 20'333'899 data points, which is about 235 days of collected data. The data set is available as CSV-files at <http://www.cse.chalmers.se/~almgren/datasets>. Figure 4.8 describes the structure of the data set.

²<https://www.backblaze.com/blog/the-3-2-1-backup-strategy/>

4.3.1 Household Data

The household data was gathered from every day usage of domestic fridges and summarised in Table 4.2. The data was recorded from 5 households, over several weeks each. The Z-Wave system (Section 4.1.4) was used to collect data from households, as it has more sensors, and is easier to scale up than the CLM1000 Professional system (Section 4.1.3). Appliance energy consumption was recorded for every household. For some households door openings, and temperature was also recorded.

For Household 5, a fabricated aggregated energy trace was constructed. This means that appliance level data was collected from a few high energy consuming appliances and then their traces were added together to create a fabricated energy trace. The fabricated aggregate energy trace consists of three appliances, a fridge, a computer and a large fan. The reason we did not collect an actual aggregate trace was because of limitations in the fuse boxes of the apartments that could be accessed, as well as limitations to the equipment that was used (Section 4.1.4).

The household data in CSV-format consists of one file per household, named "*Household_X*" (where X is the location ID number). See Figure 4.7 for an example of the CSV-files.

```
Time, Power[w], Location Id, Device Id, Door[on/idle/off]
2016-03-31 08:35:21:530, 0.9, 1, 1, idle
2016-03-31 08:35:22:514, 0.9, 1, 1, idle
2016-03-31 08:35:23:529, 0.9, 1, 1, idle
```

Figure 4.7: *Household CSV file*

The following is a description of each household:

Household 1 is a small student apartment with one inhabitant. The refrigerator is a combined refrigerator and freezer, of model Electrolux ERB 8445. In this household door openings and temperature was measured in addition to the energy consumption. The door opening sensor and temperature sensor were only connected to the refrigerator, and not to the freezer.

Household 2 is a small apartment with one inhabitant. The refrigerator is a combined refrigerator and freezer, of model Whirlpool WBE3321. Both temperature and door openings were recorded in addition to energy consumption. The door openings of the freezer was neglected, similarly to *Household 1*.

Households 3-5 are the same kind of student apartment as *Household 1*, but with different persons living there. The usage patterns of the fridges will therefore differ. The refrigerator is a combined refrigerator and freezer, of model Electrolux ERB 8445. Only power consumption was measured in these households.

Table 4.2: Household summary. Displaying which system is used, resolution (Hz), if aggregate consumption was collected, if door openings and temperature were measured.

| Household | System | Hz | Aggregate | Door Event | Thermometer |
|-----------|--------|----|-----------|------------|-------------|
| 1 | Z-Wave | 1 | Yes | Yes | Yes |
| 2 | Z-Wave | 1 | No | Yes | Yes |
| 3 | Z-Wave | 1 | No | No | No |
| 4 | Z-Wave | 1 | No | No | No |
| 5 | Z-Wave | 1 | No | No | No |

4.3.2 Controlled Test Environment

To make the gathered data set as diverse and rich as possible, a controlled setup environment was constructed. This environment was then disturbed to create certain patterns in the energy trace. For example the door was opened every 15 minutes, to see how the refrigerator responded. The purpose of creating these patterns were to be able to recognise the events that caused them. Either in a generic refrigerator trace or in the total household energy trace.

The test environment consists of a single smaller wine refrigerator, of model 85W CLimadiff CLS33A, connected to the CLM1000 Professional system. The CLM1000 Professional was selected because of its precise energy readings. Each experiment was carried out in a time frame of 8 hours and a week to make sure that the characteristics of each experiment shows properly in the energy trace of the refrigerator.

The following five tests were performed:

- 1. Empty refrigerator with door closed:** The empty refrigerator works as a baseline for all the other test cases. This is considered the normal state of the refrigerator.
- 2. Full refrigerator with door closed:** The refrigerator was filled with bottles containing water. The energy consumption was recorded when the temperature of the refrigerator and the containers had stabilised.
- 3. Empty refrigerator with frequent openings:** The door was opened every 15 minutes for a couple of seconds. Long enough for cold air to escape and warm air to enter.
- 4. Refrigerator filled with hot containers:** The refrigerator was filled with bottles containing hot water. This was done when the fridge was in a stabilised state, where it had not been open or tampered with for some time.

The test environment CSV-format data consists of one file per test. The files are named after abbreviation of Controlled Test Environment, CTE, and the number

from the list above corresponding to each test. Example of file name: cte_1.csv is the first test in the list above.

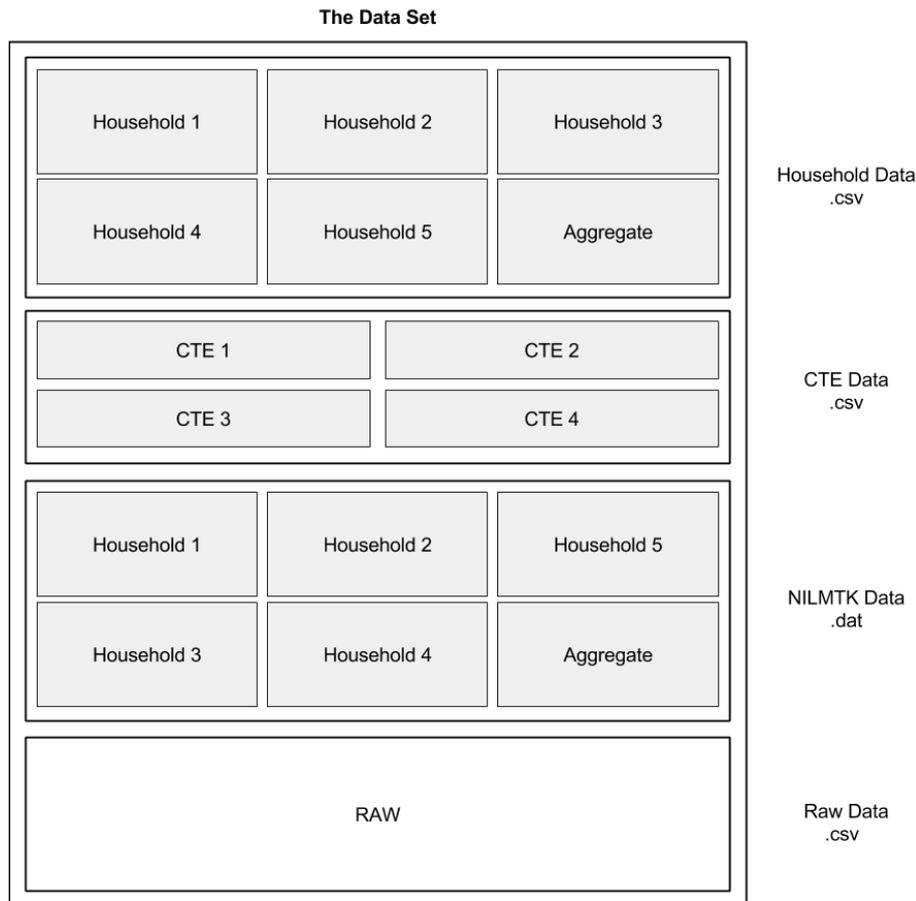


Figure 4.8: *The structure of our data set.*

4.3.3 Analysis Toolkit Data

The NILMTK data is data taken from the Household Data and then formatted to work with the NILM tool kit. The data consists of only energy data from the five households, and the fabricated aggregate data (Section 4.3.1). This data is stored in .dat files, with only time-stamps and power data.

4.3.4 Raw Data

The raw data is all the data gathered throughout the whole master thesis project, including the data above. This data include data that was gathered while testing the collection system.

It also contains data from households where more data was gathered than for other households. We wanted household data and NILMTK-data to contain the

same amount of data points. This data is not formatted in any way, but just saved in CSV-files.

5

Data Set Analysis

After the data set was gathered, different methods of analysis was applied to review the quality of the data set. Mainly three methods were used: Visual Analysis was performed, comparisons were made to data sets found in previous work, and finally disaggregation algorithms were applied.

Visual analysis involves plotting graphs of both the natural household data and the controlled environment tests to review the general quality of the data set and see if the different households provided any unforeseen results.

For data set comparison, we reviewed data sets gathered in previous work and compared those data sets to ours in terms of quality.

Disaggregation algorithms were applied to the data set using the *Non-Intrusive load monitoring toolkit (NILMTK)*. Here, we present more technical information about the toolkit and how it was used by us.

5.1 Visual Analysis

To evaluate the overall quality of the data set, some visual analysis was performed.

Visual analysis is a good way to reveal large errors in the data set in order to evaluate how well the devices used to measure plug-level data worked. Missing data points or data points with unreasonably large/small values are also easily spotted.

Furthermore, visual analysis can also be used as a way of recognising patterns in the data, and especially when it comes to the controlled environment tests, for further use with *NILM* algorithms.

5.2 Data Set Comparisons

Analysis is done by comparing the collected data set to previously collected data sets in terms of content and presentation. Since the data set gathered and presented in this thesis is aimed to be lighter than most other data sets presented in previous work, much of the comparisons will conclude in extensions that can be made for future work.

Comparing the data set gathered in this thesis to previous data sets, resolutions and what data is present have been looked at. For example, *ECO* only presents data points with watts, while *REDD* also presents unix time stamps for each data point.

5.3 Analysis Toolkit NILMTK

To test disaggregation algorithms on the data set, the Non-Intrusive Load Monitoring Toolkit, *NILMTK*, was used. NILMTK is a toolkit written in Python designed to help evaluate the accuracy of NILM algorithms on available data sets as described in Chapter 2.2.

NILMTK implements four different disaggregation algorithms; The two algorithms as described in the paper by Batra et al. [5]: *Combinatorial Optimisation* and *Factorial Hidden Markov Model (FHMM)*, and two more algorithms: *Maximum Likelihood Estimation*, and an implementation of Harts disaggregation algorithm [13].

NILMTK contains a processing pipeline that can handle chunks of data. This allows the toolkit both to load arbitrarily large data sets by processing parts of it, and calculate statistics on parts of the data set at a time (data per day, data per week, etc). These pre-processing methods can also be used, for example, to fill gaps in the energy trace by re-sampling the signal.

All documentation¹ and installation guides² for NILMTK are available on the public Github.

NILMTK uses the *HDF5 (Hierarchical Data Format)* binary file format to store power- and metadata from the data sets and use it with the implemented disaggregation algorithms. HDF5 is a file format to organise and store large amounts of data in simple binary files. An example of the HDF5 structure used by NILMTK can be seen on the NILMTK Github.³

To use a data set with NILMTK, a parser must be developed between the data sets' format and the HDF5 format. NILMTK contains converters for 8 data sets to HDF5. For the data set gathered in this thesis, a balance between developing a new parser specifically for this data set, and making sure the data were similar to some previous data set with an already existing parser had to be considered.

¹<https://github.com/nilmtn/nilmtn/tree/master/docs/manual>

²https://github.com/nilmtn/nilmtn/tree/master/docs/manual/user_guide/install.md

³https://github.com/nilmtn/nilmtn/blob/master/docs/manual/user_guide/data.ipynb

6

Results

In this chapter, results are presented in the areas of building and gathering the data set, and in the area of analysing the data set.

The data set, the process of gathering it, and its analysis are all evaluated in regards to the problem statements and goals as presented in Chapter 1.1.

6.1 Data Gathering

All the systems used for data gathering have been rated according to the evaluation qualities presented in Section 3.3. The qualities of each data gathering system were rated with regards to how the system compared to the other two systems. The ratings are summarised in Table 6.1 and discussed in Chapter 10.

6.1.1 Grades

Every system will be evaluated and given a grade between 1 and 5. A grade of 1 is very bad, and enough to not recommend the system due to its grade. A five however is a top grade and the system could be considered even if lacking in other areas. Following is a short description of the grades for each design goal. The design goals are described in Chapter 3.3.

Price

A low grade is an expensive device. A high grade is a cheap device.

An exact price for what is feasible or not is hard to estimate, as the price needs to be compared to both the other qualities of the same device as well as the price of other devices and their quality. A cheap device with bad quality data still needs to be fairly compared to an expensive device with high quality data.

Availability

A low grade means that the system is very hard to obtain. For example you have to order it from some obscure website from the other side of the globe. A high grade means you can find them in any large electronic store.

Logging speed/Resolution

A low grade means that the system lacks in logging speed. Anything below three can be considered to log slower than once every 30 seconds. A high grade means fast logging speed, and a five would be logging speeds faster than once every second.

Data Quality

The data quality is related to the logging speeds, but also include for example packet loss in wireless systems. A high grade means consistent data of high quality. A low grade means sporadic data with a lot of gaps and lost data.

Versatility

A high grade means that the system has a lot of sensors that can measure different aspects of the appliance. A low grade means the opposite. For example a one would not have any sensors and the system would not be compatible with any other system. A five would mean a lot of sensors in the system as well as compatibility with other systems.

Prior Knowledge

A low grade means you have to be an expert in a certain field to understand the system. A high grade means you do not need any prior knowledge and can set up the system with just a simple guide.

Set up

A low grade reflects a high set up-time from buying the device until it is up and running, with a lot of troubleshooting. A high grade reflects a low time, and is almost plug and play.

6.1.2 Z-Wave System

Below follows a review of the Z-Wave system according to the design goals in Section 3.3, with an explanation for each of the goals on how they were given their grade.

Price: As Z-Wave is a Raspberry Pi-based system, the original setup cost is fairly cheap. To read just energy data using the Z-Wave system, the following is needed: a Raspberry Pi equipped with the RaZberry communication chip, and a wall plug sensor. This cost is approximately 1 500 SEK, and any extension sensor cost approximately 500 SEK a piece.

Availability: Z-Wave is one of the most commonly used home automation systems on the market and can be found in any large electronic store.

Logging speed/Resolution: The logging speed of the Z-Wave system can be set at a sensor level. However different sensors have different maximum logging speeds. The energy monitor wall plug has a maximum logging speed of 1 Hz.

Data Quality: The data quality of the Z-Wave system is overall good. However at a logging speed of 1 Hz, sometimes the sensors glitch and send approximately 5 values within 1 second, and nothing the next 4 seconds. This is not a major problem as it is not very common and the timestamps differ at micro second level.

Versatility: The Z-Wave system has access to many different sensors, and no extra work needs to be put into getting these sensors to work with the system.

Prior Knowledge: The Z-Wave system needs no prior knowledge to use out of the box and can be used entirely with the built in web GUI. However to use it as has been done during this thesis a basic understanding of programming and preferably Python scripting is needed.

Set up: The Z-Wave system made it easy to set up each individual logging device, and the Z-Way web interface provided easy access to each device as well as a good overview of the system. It is also easy to introduce new sensors into the system.

6.1.3 Open Energy Monitor

Below follows a review of the Open Energy Monitor system according to the design goals in Section 3.3, with an explanation for each of the goals how they were given their grade.

Price: A base setup with a Raspberry Pi, their receiver chip and transmitter unit costs approximately 1 000 SEK, with every sensor costing approximately an additional 100 SEK.

This price is low considering the budget of an ordinary research project of this kind. The initial price of 1 000 SEK is also low considering the low price of adding more sensors to the setup.

Availability: The Open Energy Monitor can only be purchased from the official OEM store.¹ It ships from the UK and takes a few work days to arrive.

In a project where sensors might break down and it is crucial to be able to replace and extend the project with additional sensors, the availability of the OEM may be a reason not to consider it as a data gathering tool for a project.

Logging speed/Resolution: The Open Energy Monitor logs data at 1 Hz, which is satisfactory when looking at the requirements of analytics algorithms.

¹<https://shop.openenergymonitor.com/home-energy-monitor/>

Data Quality: As problems was encountered getting the Open Energy Monitor system to work, the data quality produced by it cannot be reviewed.

However, after investigating the technical specifications of the Open Energy Monitor system, it seems that it is able to produce high quality data and would be a suitable candidate for producing a data set if one could get past all technical set-up issues.

Versatility: This system have more compatible sensors beyond standard energy sensors, such as temperature sensors and humidity sensors. However none of these were tested in this project.

Prior Knowledge: The open energy monitor requires some background in electrical engineering to get it working properly. Many concepts involving the open energy monitor were to us as computer science students often very foreign.

As the OEM tool was not able to be used for this project, we feel that the overhead that comes with learning the tool and how it works is not worth it in the long run as the Z-Wave system does the same measurements with a lower learning overhead.

Set up: The Open Energy Monitor System had the most complex set up of all the evaluated systems. It requires extensive electrical know how and when we were told on support forums to start removing resistors from the transmitter to get our version to work with the new code, we decided to not pursue this particular system for data collection.

It also does not work with multi-carrying cords, which almost all home appliance use. To work around this, some sort of adapter needs to be constructed.

6.1.4 CLM1000 Professional

Below follows a review of the CLM1000 according to the design goals in Section 3.3, with an explanation for each of the goals how they were given their grade.

Price: The CLM1000 costs approximately 1 800 SEK² and is available commercially in Sweden. The device is quite expensive, but also produces very high quality data, and thus a trade-off has to be made between price and quality. To setup a similar system as in this thesis, the CLM1000 also needs a Raspberry Pi, which cost around 500 SEK.

Availability: CLM1000 Professional Can be found on several different websites, but is not very common in regular electronic stores.

Logging speed/Resolution: The CLM1000 Professional is logging detailed data at 1 Hz without any problems. The data received from the device sent by serial port

²<https://www.elfa.se/en/output-measuring-device-type-cee-christ-elektronik-clm1000-standard/p/17648131>

Table 6.1: Summary of the grades for all systems.

| | Z-Wave System | Open Energy Monitor | CLM1000 Professional |
|------------------------|----------------------|----------------------------|-----------------------------|
| Price | 4 | 5 | 1 |
| Availability | 5 | 3 | 3 |
| Logging Speed | 4 | 4 | 4 |
| Data Quality | 4 | 1 | 5 |
| Versatility | 5 | 4 | 2 |
| Prior Knowledge | 3 | 1 | 4 |
| Setup Time | 4 | 1 | 5 |
| Average | 4.14 | 2,71 | 3.43 |

to a computer is structured nicely with clear formatting making it very simple to parse and upload to the database.

Data Quality: The energy data produced by the CLM1000 Professional is of very high quality with many different parameters logged (Section 4.1.3).

Versatility: The CLM1000 Professional is only a power reader and needs sensors from another system to measure anything more than just power. As it can be connected to a Raspberry Pi it can be combined with one of the other systems we tried to add more sensors.

Prior Knowledge: To use the CLM1000 Professional as an out-of-the-box solution, no prior knowledge whatsoever is required. For automatising the CLM1000 Professional, minor programming knowledge is required. One must know how to read serial port data using some piece of code, which is not very difficult. We provide a small piece of Python code for this purpose (Appendix A.1).

Set up: The setup of the CLM1000 Professional is easy, however the energy reader is rather big and has to be connected to a Raspberry Pi for data transfer. This requires space and is not ideal for being placed close to an appliance in an ordinary home. For that reason it is more suitable for test purposes than actual deployment.

6.2 Analysis

In this section results from performing visual analysis, from comparing the gathered data set with other previously gathered data sets, and from using *NILMTK* to test Non-Intrusive Load Monitoring algorithms on the data set are presented.

6.2.1 Visual Analysis

To get a better understanding of how the patterns from the refrigerator looked like, some visual analysis was performed. The energy trace from appliances in the data

6. Results

set was plotted. By looking at the graphs created, it could be determined how different patterns appeared. This was mainly used to look at the output from the controlled test environment. The results are presented in Figures 6.1 - 6.4.

Figure 6.1 is the energy trace of an empty refrigerator, where the door has been kept shut. You can clearly see the characteristic spike when the refrigerator turns on, and also when the refrigerator turns back off.

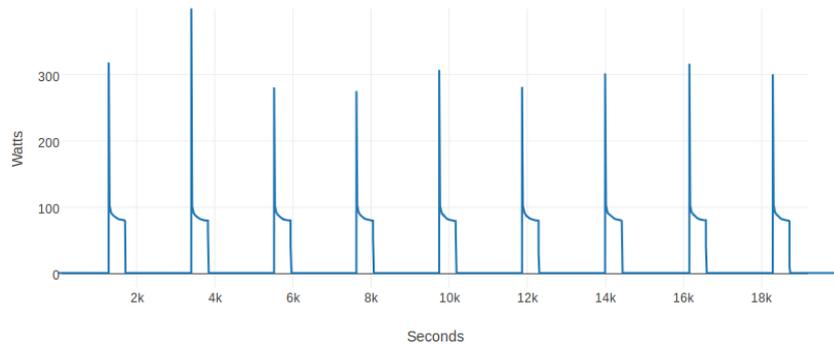


Figure 6.1: The energy trace of a closed empty refrigerator.

Figure 6.2 is a comparison of the empty refrigerator with the door shut to an empty refrigerator with frequent door openings. A difference in frequency of energy trace spikes is clearly visible.

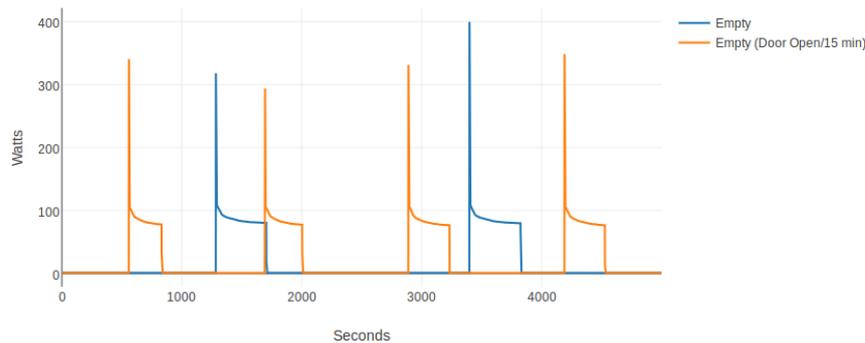


Figure 6.2: *The energy trace of a closed empty refrigerator compared to the energy trace of a refrigerator with frequent door openings.*

Figure 6.3 shows the difference between an empty refrigerator and a refrigerator with hot containers of water placed inside. The trace of the filled refrigerator stays on for a longer period of time.

Figure 6.4 shows the energy trace of a empty refrigerator, just as it is being filled with hot containers of water. The refrigerator stays on for a noticeable longer time

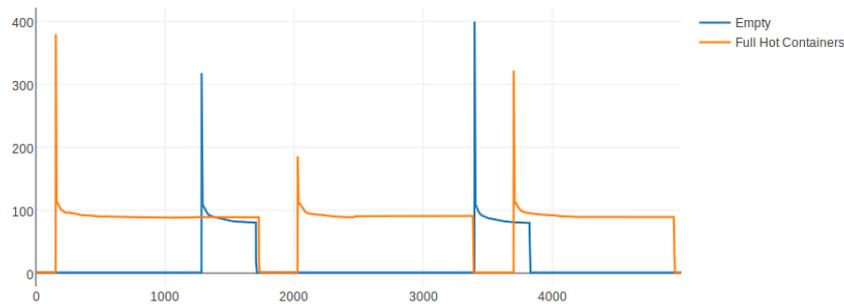


Figure 6.3: *The energy trace of a closed empty refrigerator compared to the energy trace of a refrigerator with hot containers placed inside.*

in the beginning as it is cooling down the hot water, but eventually stabilises back to its normal temperature.

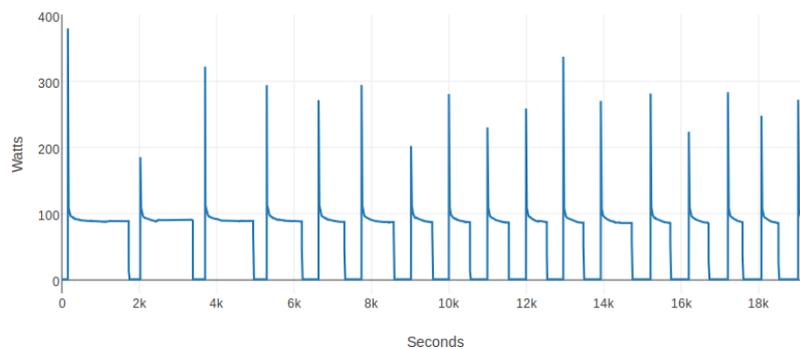


Figure 6.4: *A empty refrigerator as it is being filled with hot containers.*

Figure 6.5 compares two different models of refrigerators. The orange line shows an older refrigerator while the blue line shows a newer model. The older fridges stays on for much longer than the newer fridge, however the new fridge turns on more often. This probably means that the newer fridge is much more efficient at cooling the space inside the fridge.

We can by looking at these graphs conclude that a satisfactory quality of data was gathered. The different characteristics of the controlled tests are easily spotted in the plotted graphs. The characteristic spikes of the refrigerators are also being captured in a good way and no spike is missed.

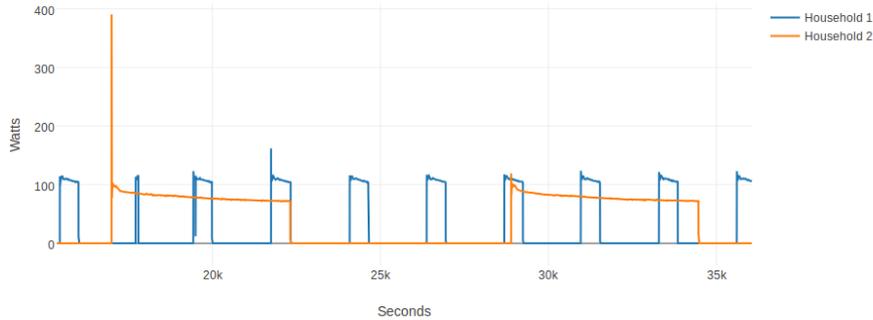


Figure 6.5: *A empty refrigerator as it is being filled with hot containers.*

6.2.2 Data Set Comparisons

Most other data sets such as *ECO*, *REDD*, etc gather plug-level data at a rate of 1 Hz. We have successfully matched this frequency and are yielding good results with both the Z-Wave System setup and the CLM1000 Professional setup.

Other data sets use complicated CT clamp-based setups for measuring the aggregated household trace and circuit-level data. We instead use an optic sensor to measure the aggregated energy trace of a household through the smart meter installed in the home. This makes the data gathering process easier to perform, even though the results might not be as precise. For projects with the goal of large deployment, it might be a better solution.

6.2.3 Analysis Through NILMTK

NILMTK provides easy tools for presenting both aggregated energy traces as well as individual appliance energy traces. The toolkit also includes functionality to split the data set at given timestamps to provide the NILM algorithms with training data.

We were able to run some of the NILMTK functionality on our data set. Providing it with our fabricated aggregated energy trace, described in Section 4.3.1, we could for example see which appliance was consuming the most energy.

However when attempting to disaggregate the data set, no analytics functions could be ran to get metrics for how well the disaggregation worked. The success of the disaggregation is measured by a F-score. It is unclear why this did not work. It might have been due to the fact our data set is too small or that we fabricated the total aggregate. Investigation and analysis were carried out with creators and contributors of NILMTK without success.

7

Ethics

Here, ethical questions that arose both during the research part of the project as well as during the creation of our data set is presented.

The ethical questions have been divided into two parts: Surveillance and Sustainability. The Surveillance covers how energy trace data in conjunction with analysis can intrude on peoples privacy, and how it should be used to avoid such intrusions. Sustainability covers if these kinds of systems are fair to use to further contribute to a greener world.

7.1 Surveillance

In its original form, energy data itself is not intrusive, but it all boils down to how the data is used. With good enough analytics tools and algorithms, very precise information about when and how appliances in a data set are used can be deducted from aggregated energy traces.

It therefore falls in the hands of the developers and owners of the data set to clearly communicate to inhabitants of households how their energy data can and will be used. As is with many other ethical questions, the solution is often clarity and transparency. If all parties are aware of how the data is used, there will be fewer problems. Tools for analysing data sets should to the greatest extent possible be kept open-source for even further transparency.

When creating the data set for this thesis, no information about the inhabitants of the households were stored except for technical information about the devices and meters used. We also plan for the future that any extensions to the data set shows the same respect for the inhabitants' privacy.

7.2 Sustainability

A big motivation for conducting projects in the area of appliance energy traces is to make the general public more aware of their energy footprint. After applying analytics to the energy data, more appliance-specific information can be revealed and it becomes easier to get a detailed overview of the energy trace.

This information can then help the public find what appliances in their household uses the most energy, and either help them replace them with a better device, or

help them use their device as energy smart as possible.

The meters installed to measure the energy trace of appliances and entire households in this project are all very energy efficient and the changes in the total energy consumption is thus barely noticeable.

The software required to operate the plugs and log the data is very light-weight and can basically be run on any system with an internet connection. Because of this, no extensive hardware is required for the system, or the system can even be run on already existing hardware such as home computers and media servers.

8

Lessons Learnt

In this chapter, some of the more prominent issues that was faced during the development of the data set are presented. This is done in order to assist future efforts in locating and avoiding such issues.

For each area of issues, some thoughts and suggestions on solving these problems are also presented. The issues range from problems encountered during research to problems encountered when implementing and testing the software setup.

8.1 Hardware

Hardware crashes can always be an issue, as it is often harder to notice than a software malfunction. If a data gathering tool is supposed to be evaluated in the same way as in this thesis, it is recommended to always buy multiple units of the same model as to quickly eliminate the cause of an error; if one device is failing and a software problem cannot be located, the other unit can always be tested.

8.2 The Data Set

When the data set starts to grow in size, it can be somewhat hard to get a good overview of what it actually contains. Unless good tools are in place in beforehand to view the data set in a comprehensive way, it can be difficult to see for example if parts of the data is bad, if a script has crashed somewhere, or if a power meter is starting to fail.

In the start of the project, database plotting tools such as Plotly¹ was used to draw graphs, but it started to have problems displaying the data as the set grew larger. After the *NILMTK* toolkit was up and running, it did a much better job at plotting the data to give a better overview.

8.3 Software

When developing scripts whose purpose is to run for longer period of time, it is necessary to set up proper logging tools to notice crashes and make sure they are

¹<https://plot.ly/>

handled correctly.

When a software or hardware malfunction (script reading wrong data, hardware sending the wrong data, etc) happens and your uploading scripts do not crash or report an error, you will get faulty data in your database. This data can be either completely unreasonable values or empty values, or in the worst case; the data reassembles real data.

Thus, if no logging tool is set up, or it is set up improperly to handle a certain kind of malfunction, you can lose a lot of data points in the process since it may prove difficult to see where the crash occurred and what data has been infected.

We recommend some alert-base system where the user can receive notifications whenever a script behaves badly or crashes, in order to avoid unnecessary problems.

9

Extensions

Here, possible extensions to the thesis are presented to inspire future aspirants in the field of research of energy trace data sets. There are several areas in which the project could be extended: In data gathering, in how the data set is formed, and in how the data is analysed.

9.1 The Data Set

The most obvious extension of this master thesis would be to start collecting data from more appliances than just refrigerators. Collecting data from all appliances in a household would give a more complete picture of the total energy trace of a household and would aid in future non-intrusive load monitoring efforts.

More data could also be gathered in the areas of mains and circuit-level data. Then, more information could be said not only about each device, but also about the room or the part of the household that the device resides in.

9.2 Refrigerator Model

If refrigerators are to stay the main focus of the data set, there are a lot of extensions that could be made in terms of sensors. Laguerre et al. [19] present a model of a refrigerator including air flow, heat transfer, water evaporation and water condensation. More parameters like these could possibly provide for a more detailed model of the refrigerator and correlations between events and the energy trace could be easier to locate with NILM algorithms.

9.3 Analysis

Due to our limited background in machine learning, the machine learning analysis methods were treated as black boxes. However, machine learning algorithms and especially NILM-algorithms is a perfect area to extend the project into. A good place to start is trying to modify the NILM-algorithms to be able to recognise the patterns from the controlled test environments, and thus be able to label not only a certain device in the aggregated trace, but also certain events of that device in the trace.

9.4 Alternative use of Energy Traces

The behaviour of inhabitants of a household can be mapped by analysing the energy trace from said household. An interesting extension of this master thesis could be to analyse other types of energy traces to see what could be monitored. Within computer science an interesting area would be to analyse the energy trace of an active computer and try to draw conclusions of what applications the user is currently using. [9]

10

Conclusion

In this thesis, different methods to collect energy trace data from households and refrigerators were proposed and evaluated. Three methods were chosen, and a data set was gathered from a small number of refrigerators, both in controlled environments and in actual households.

Three setups for data gathering were researched: The Z-Wave system, the CLM1000 Professional, and the Open Energy Monitor system. The Z-Wave system and CLM1000 run Python scripts to log data from sensors and send it to a database. The Open Energy Monitor system implementation had to be abandoned due to technical difficulties and time constraints, but was still discussed and evaluated.

Z-Wave system became the system of choice to gather household data. The system is fairly cheap, with an intuitive interface and good scaling potential. The Open Energy Monitor system has potential, but requires a lot of background knowledge to set up and get it to work properly.

The data set gathered is of similar quality to previous data sets, even though it lacks somewhat in quantity. However the same quantity of data can be obtained by collecting data over a longer period of time. The data set consists at the time of writing this paper of household data from 5 households and a total of 20'333'900 data points, which is roughly 235 days.

Upon initiating the thesis, four goals were set up for retrospectively judging the success of the work. These four goals were: Investigation, Data Gathering Platform Creation, Data Set Gathering, and finally Evaluation. The goals are explained more in-depth in Chapter 1.1.

Investigation was carried out according to goal 1. This was done by researching past work in the field and presenting it as previous work, and by then using this knowledge to develop our own methods.

An energy trace gathering system was constructed as per goal 2. For this purpose, some software had to be implemented.

The data set was gathered using the systems/platforms chosen from achieving goal 2, according to goal 3.

Finally, the gathered data set was evaluated by comparing it to previous data sets and by reasoning how well it works with existing NILM-algorithms and tool kits.

Overall, all goals were met and a solid foundation for future efforts were built, both in the area of household energy trace research, as well as in the area of NILM-

10. Conclusion

algorithm application research. The research presented can be used for future efforts wanting to gather data, The software and hardware implemented and utilised can be used to extend the data set or create new data entirely, the algorithms and methods used to analyse the data set is freely available and applicable to multiple sources of data.

Bibliography

- [1] K. Anderson, A. Ocneanu, D. Benitez, D. Carlson, A. Rowe, and M. Berges. BLUED: A fully labeled public dataset for event-based non-intrusive load monitoring research. In *Proceedings of the 2nd KDD workshop on data mining applications in sustainability (SustKDD)*, pages 1–5, 2012.
- [2] M. Baranski and J. Voss. Genetic algorithm for pattern detection in NIALM systems. In *Systems, man and cybernetics, 2004 IEEE international conference on*, volume 4, pages 3462–3468. IEEE, 2004.
- [3] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy, and J. Albrecht. Smart*: An open data set and tools for enabling research in sustainable homes. *SustKDD, August*, 111:112, 2012.
- [4] N. Batra, M. Gulati, A. Singh, and M. B. Srivastava. It’s Different: Insights into home energy consumption in India. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, pages 1–8. ACM, 2013.
- [5] N. Batra, J. Kelly, O. Parson, H. Dutta, W. Knottenbelt, A. Rogers, A. Singh, and M. Srivastava. NILMTK: an open source toolkit for non-intrusive load monitoring. In *Proceedings of the 5th international conference on Future energy systems*, pages 265–276. ACM, 2014.
- [6] C. Beckel, W. Kleiminger, R. Cicchetti, T. Staake, and S. Santini. The ECO Data Set and the Performance of Non-Intrusive Load Monitoring Algorithms. In *Proceedings of the 1st ACM International Conference on Embedded Systems for Energy-Efficient Buildings (BuildSys 2014). Memphis, TN, USA*, pages 80–89. ACM, Nov. 2014.
- [7] A. Boicea, F. Radulescu, and L. I. Agapin. MongoDB vs Oracle–Database Comparison. In *2012 Third International Conference on Emerging Intelligent Data and Web Technologies*, pages 330–335. IEEE, 2012.
- [8] M. Chui, M. Löffler, and R. Roberts. The Internet of Things. *McKinsey Quarterly*, 2(2010):1–9, 2010.
- [9] S. S. Clark, H. Mustafa, B. Ransford, J. Sorber, K. Fu, and W. Xu. Current events: Identifying webpages by tapping the electrical outlet. In *European Symposium on Research in Computer Security*, pages 700–717. Springer, 2013.

- [10] M. R. Garey and D. S. Johnson. *Computers and Intractability*, volume 29. WH Freeman and Company, New York, 2002.
- [11] Z. Ghahramani and M. I. Jordan. Factorial Hidden Markov Models. *Machine Learning*, 29(2):245–273, 1997.
- [12] J. Gupta, M. R. Gopal, and S. Chakraborty. Modeling of a domestic frost-free refrigerator. *International Journal of Refrigeration*, 30(2):311 – 322, 2007.
- [13] G. W. Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.
- [14] C. Holcomb. Pecan Street Inc.: A test-bed for NILM. In *International Workshop on Non-Intrusive Load Monitoring, Pittsburgh, PA, USA*, 2012.
- [15] W. Kleiminger, C. Beckel, T. Staake, and S. Santini. Occupancy detection from electricity consumption data. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, BuildSys’13, pages 10:1–10:8, New York, NY, USA, 2013. ACM.
- [16] J. Z. Kolter and T. Jaakkola. Approximate inference in additive factorial HMMS with application to energy disaggregation. In *International conference on artificial intelligence and statistics*, pages 1472–1482, 2012.
- [17] J. Z. Kolter and M. J. Johnson. REDD: A public data set for energy disaggregation research. In *Workshop on Data Mining Applications in Sustainability (SIGKDD)*, San Diego, CA, volume 25, pages 59–62, 2011.
- [18] O. Laguerre, S. Benamara, and D. Flick. Numerical simulation of simultaneous heat and moisture transfer in a domestic refrigerator. *International journal of refrigeration*, 33(7):1425–1433, 2010.
- [19] O. Laguerre, S. Benamara, and D. Flick. Numerical simulation of simultaneous heat and moisture transfer in a domestic refrigerator. *International journal of refrigeration*, 33(7):1425–1433, 2010.
- [20] S. Makonin, F. Popowich, L. Bartram, B. Gill, and I. V. Bajić. AMPDS: A Public Dataset for Load Disaggregation and Eco-Feedback Research. In *Electrical Power Energy Conference (EPEC), 2013 IEEE*, pages 1–6, Aug 2013.
- [21] O. Parson, S. Ghosh, M. Weal, and A. Rogers. Non-intrusive load monitoring using prior models of general appliance types. *Proceedings of the Twenty-Sixth Conference on Artificial Intelligence (AAAI-12)*, pages 356–362, 2012.
- [22] USA Energy Information Administration. Assessment of consumption and expenditure data collected from energy suppliers against bill data obtained from interviewed households: Case study with 2009 recs. www.eia.gov, 2013. Accessed 2016-11-02.
- [23] A. J. Viterbi. Error bounds for convolutional codes and asymptotically optimum decoding algorithm. *Proc. PerCom*, 2012.

- [24] M. Weiss, A. Helfenstein, F. Mattern, and T. Staake. Leveraging smart meter data to recognize home appliances. In *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, pages 190–197. IEEE, 2012.
- [25] J.-P. Zimmermann, M. Evans, J. Griggs, N. King, L. Harding, P. Roberts, and C. Evans. Household electricity survey: A study of domestic electrical product usage. *Intertek Testing & Certification Ltd*, 2012.

A

Appendix 1

A.1 CLM1000 Professional Automatisation Script

```
1 import serial
2
3 # Change ttyUSB0 to whatever USB port the CLM is connected to
4 ser = serial.Serial('/dev/ttyUSB0', 115200, timeout=1)
5 # Can also be used without while loop in crontab
6 while(1):
7     bytesToRead = ser.inWaiting()
8     s = ser.read(bytesToRead)
9     data = str(s)
10    with open("data.txt", "a") as myfile:
11        myfile.write(data)
12
13 ser.close()
```

Listing 1: A small script to automatically read serial port data from the CLM1000 Professional. Works on a Raspberry Pi.

A.2 CLM1000 Professional Upload Script

```
1 content = []
2 with open('init.txt') as my_file:
3     content = my_file.read().splitlines()
4     # Read database info from content
5     #Can also be used without while loop in crontab
6     while(1):
7         con = mdb.connect(ip,login,password,database)
8         with open(sys.argv[1],'r') as my_file:
9             for line in my_file:
10                temp = line.split(';')
11                if len(temp[0]) == 19 and len(temp) == 12:
12                    s = temp[0].replace(" ",":")
13                    timer = time.mktime(
14                        datetime.datetime.strptime(
15                            s,"%d.%m.%Y:%H:%M:%S").timetuple())
16                    # Insert SQL query here, values are found in temp.
17                    # sql =
18                    try:
19                        cur = con.cursor()
20                        cur.execute(sql)
21                    except mdb.Error, e:
22                        print "Error %d: %s" % (e.args[0],e.args[1])
23                        sys.exit(1)
24                else:
25                    print "Not proper formatting of input"
26            con.commit()
27            cur.close()
28            con.close()
29            open(sys.argv[1], 'w').close()
30            time.sleep(600)
```

Listing 2: A simple upload script for the CLM1000. Reads the data saved by the automatising script and uploads it into the MySQL database.

A.3 Z-Wave Upload Script

```

1 while(1):
2     #Reads the log files and creates an entry in the mongodict for every second.
3     for arg in files:
4         print "Processing New file: " + arg
5         # Check if they have content, otherwise print error and break for loop
6         if os.stat(arg).st_size > 0:
7             with open(arg,'r+') as my_file:
8                 datas = json.load(my_file)
9                 # Iterate through the sensorData part of json object
10                for data in datas["sensorData"]:
11                    # If the timestamp has not been entered previously to dict
12                    if int(data['time']) not in mongodict:
13                        point = Data_point()
14                        if arg == file_w1:
15                            point.w1 = data['value']
16                        elif arg == file_d:
17                            point.d = data['value']
18                        elif arg == file_t:
19                            point.t = data['value']
20                        mongodict[int(data['time'])] = point
21                        timestamps.append(int(data['time']))
22                    # Otherwise
23                    else:
24                        temppoint = mongodict[int(data['time'])]
25                        if arg == file_w1:
26                            temppoint.w1 = data['value']
27                        elif arg == file_d:
28                            temppoint.d = data['value']
29                        elif arg == file_t:
30                            temppoint.t = data['value']
31                        mongodict[int(data['time'])] = temppoint
32                    # Truncate the file just read
33                    open(arg, 'w').close()

```

Listing 3: A simple upload script for the Z-Wave. Reads data from json files found on the Raspberry and saves in a dictionary to be uploaded to a database. Full script found on Github.

A.4 Raspberry Pi Hardware Spec

The Raspberry Pi 2 Model B has the following Hardware specifications:

- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port
- Combined 3.5mm audio jack and composite video
- Camera interface (CSI)
- Display interface (DSI)
- Micro SD card slot
- VideoCore IV 3D graphics core

Z-Wave provides a ready-to-go image complete with Raspbian and all software needed for running the Z-Wave system:

<http://rasberry.z-wave.me/index.php?id=24>

A.5 Z-Wave Hardware Spec

Information about the Raspberry chip used to communicate with Z-Wave sensors can be found on the Z-Wave website.¹

Information about different sensors for the Z-Wave system:

Aeon Labs ZW075-C16: <http://products.z-wavealliance.org/products/1045>

Fibaro Wall Plug: <http://www.fibaro.com/en/the-fibaro-system/wall-plug>

Fibaro Door/Window Sensor: <http://www.fibaro.com/en/the-fibaro-system/door-window-sensor>

DS18B20 Digital Thermometer: <https://cdn-shop.adafruit.com/datasheets/DS18B20.pdf>

A.6 Open Energy Monitor Hardware Spec

Project Website: <https://openenergymonitor.org/emon/>

EmonTx Specification: https://wiki.openenergymonitor.org/index.php/EmonTx_V3

EmonPi Software: <https://github.com/openenergymonitor/emonpi>

A.7 CLM1000 Professional Hardware Specification

Hardware Specification: <http://www.christ-es.com/shop/Measuring%20devices/Power%20measuring%20devices/CLM1000-HS/>

Instruction Manual: http://www.christ-es.de/elements/products/files/26_E461757_BA_REV03_07-01-14_CLM1000-Professional_englisch_Instruction_manual.pdf

¹<http://rasberry.z-wave.me/>