The ECO Data Set and the Performance of Non-Intrusive Load Monitoring Algorithms

Christian Beckel
Dept. of Computer Science
ETH Zurich, Switzerland
beckel@inf.ethz.ch

Wilhelm Kleiminger
Dept. of Computer Science
ETH Zurich, Switzerland
kleiminger@inf.ethz.ch

Romano Cicchetti
Dept. of Computer Science
ETH Zurich, Switzerland
cicchetti@student.ethz.ch

Thorsten Staake
Energy Efficient Systems
University of Bamberg, Germany
thorsten.staake@uni-bamberg.de

Silvia Santini
Wireless Sensor Networks Lab
TU Darmstadt, Germany
santinis@wsn.tu-darmstadt.de

Abstract

Non-intrusive load monitoring (NILM) is a popular approach to estimate appliance-level electricity consumption from aggregate consumption data of households. Assessing the suitability of NILM algorithms to be used in real scenarios is however still cumbersome, mainly because there exists no standardized evaluation procedure for NILM algorithms and the availability of comprehensive electricity consumption data sets on which to run such a procedure is still limited. This paper contributes to the solution of this problem by: (1) outlining the key dimensions of the design space of NILM algorithms; (2) presenting a novel, comprehensive data set to evaluate the performance of NILM algorithms; (3) describing the design and implementation of a framework that significantly eases the evaluation of NILM algorithms using different data sets and parameter configurations; (4) demonstrating the use of the presented framework and data set through an extensive performance evaluation of four selected NILM algorithms. Both the presented data set and the evaluation framework are made publicly available.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

Keywords

Smart metering, energy efficiency, non-intrusive load monitoring, evaluation framework, public data set

General Terms

Design, Algorithms, Performance, Management

BuildSys'14, November 5–6, 2014, Memphis, TN, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3143-2/14/11 ...$15.00
http://dx.doi.org/10.1145/2674061.2674064

1 Introduction

Providing feedback on electricity consumption is a powerful way to induce a more energy-efficient behavior in households [10, 12]. In particular, feedback has shown to be effective when it is detailed and provided in a timely manner, it is tailored to individual households and it contains information on the consumption of individual appliances [2, 10, 12]. Utilities, which are increasingly forced (and motivated) by policy makers to help their customers save electricity, are thus highly interested in providing appliance-specific consumption feedback as a service to their customers (e.g., in the form of automated saving recommendations). The data needed to provide such feedback could be obtained through sensors that monitor the consumption of individual appliances in the household. Deploying such a sensing infrastructure is however costly and cumbersome.

To avoid the need of monitoring individual appliances, *non-intrusive load monitoring* (NILM) algorithms have been proposed in the literature [32, 33]. These algorithms analyze the *aggregate electricity consumption* of the household, i.e. the total electricity consumption of the household measured using a single electricity meter. Through this analysis, the algorithms can identify which individual appliances are running and how much electricity they consume. NILM approaches might differ on several aspects, including the granularity at which they assume consumption data to be available or whether they apply supervised or unsupervised methods to learn typical consumption patterns of household appliances. NILM algorithms are often evaluated on single, possibly non publicly available data sets and the parameter of the algorithm are tuned to operate on those data sets [32, 33]. Different underlying assumptions, tailored parameter settings, and lack of comprehensive data sets thus make the evaluation of NILM algorithms to be often non-exhaustive but still cumbersome and time consuming. This also hampers the possibility to compare the performance of existing approaches and derive general insights about which algorithms are best suited to be used in which scenario.

In this paper, we address the problem described above and make the following contributions. First, we outline the key dimensions of the design space of these algorithms. Second, we describe a novel, comprehensive data set – the *ECO data set* (Electricity Consumption and Occupancy) – that can be used to assess the performance of NILM algorithms. While we had relied on this data set to evaluate an approach to detect household occupancy in previous work [21], we present here the data set in detail and make it publicly available.[1] With respect to other data sets, the ECO data set provides a unique combination of quality and quantity of electricity consumption data. In particular, it contains aggregate electricity consumption data – including real and reactive power for each of the three phases – and plug-level measurements of selected household appliances. The data has been collected at 1 Hz granularity and over a period of 8 months. Furthermore, the data set also contains occupancy information of the monitored households. Third, we describe the design and implementation of a comprehensive evaluation framework for NILM algorithms. The framework, called *NILM-Eval*, is similar in scope to the recently presented NILMTK framework [6] and aims at allowing researchers to run comprehensive performance evaluations of NILM algorithms. NILMTK has rich metadata support [17], preprocessing capabilities, and supports different statistics functions and performance metrics. With respect to NILMTK, NILM-Eval facilitates the design and execution of large experiments that consider several different parameter settings of various NILM algorithms. Furthermore, while NILMTK is written in Python, NILM-Eval is based on Matlab. Like for the ECO data set, we make the NILM-Eval framework publicly available.[2] The last contribution of this paper consists in the evaluation of the performance of selected NILM algorithms. The algorithms are chosen so as to represent different sectors of the design space of NILM algorithms. We evaluate their performance using our NILM-Eval framework and rely on the ECO data set. The obtained results allow to gain insights about the performance of the selected algorithms, to outline their trade-offs, and to discover potential for further improvements of the considered algorithms.

2 Design Space

The first known NILM approach has been proposed by Hart [14] in 1992. Hart's algorithm identifies step changes in the aggregate electricity consumption and matches them with a signature database to learn which appliance has been switched on or off. Building upon Hart's seminal work, several different algorithms that rely on different principles (e.g., combinatorial or probabilistic), utilize different learning methods, or rely on different data granularities have been proposed in the literature [32, 33]. Three key design parameters must however be considered when deciding which NILM algorithm to use in a real scenario: *data granularity*, *learning methods* and *information detail*.

The first dimension, *data granularity*, represents the data granularity for which the algorithms were designed and optimized for – although most of the algorithms can potentially also run on data of a different granularity. The granularity typically ranges from 1/60 Hz (i.e., data aggregated to one value per minute) [27] to multiple kilohertz (e.g., [8, 13]).

NILM algorithms may utilize different *learning methods*. There exist unsupervised and supervised NILM algorithms as well as approaches that utilize generic appliance models and can thus be classified as semi-supervised. Unsupervised approaches typically rely on low-frequency (i.e., 1 Hz) aggregate consumption data [3, 19, 22]. Baranski and Voss, for instance, detect switching events in the aggregate consumption data and use them as input to a genetic algorithm, which automatically creates event chains for different appliances [3]. Other authors utilize hidden Markov models (HMMs) to model the states of each appliance [19, 22].

Supervised approaches can be classified by the granularity of consumption data they are developed for. Gupta et al. [13], for instance, developed the algorithm *ElectriSense*, which detects consumer electronics devices and fluorescent lighting by their electromagnetic interference generated during operation. To this end, the authors rely on consumption data measured at 10 kHz. Similarly, Berges et al. rely on measurements conducted at 20 Hz to detect edges in the aggregate electricity consumption of a household using features computed on both real and reactive power [8]. Farinaccio et al. developed a pattern recognition approach that applies rules to identify the consumption pattern of a refrigerator and a heater [11]. Both Weiss et al. [31] and Marchiori et al. [25] make use of real and reactive power measurements: The former approach is closely related to Hart's algorithm as it detects switching events of appliances in the consumption pattern. The latter creates 2-dimensional histograms using the real and reactive power measurements and subsequently applies a clustering procedure to identify clusters that belong to individual appliances. Spiegel et al. [29] pursue a classification approach using features (i.e., the first order difference of the consumption data) extracted from 1 Hz real power measurements. Finally, Parson et al. [27] developed an approach based on hidden Markov models that only requires data at a granularity of 1/60 Hz. In contrast to the other approaches, Parson's algorithm is considered semi-supervised, which means it utilizes generic appliance models, avoiding the need to intrusively install sensors or use other training methods when installing the system in practice.

Finally, algorithms differ in *information detail*, which is the type of data they assume to be available. For instance, some of the aforementioned algorithms require real power consumption data only (e.g., [3], [11], [19], [22], [27], [29]). Other algorithms rely on both real and reactive power (e.g., [8], [14], [25], [31]) or make use of the fact that the consumption is split into individual phases (e.g., [31]). Other approaches utilize information provided by other sensors as additional input to the algorithm [16, 20], which can improve the estimation performance compared to analyzing the aggregate electricity consumption only. An example for such a sensor is an event detector developed by Rowe et al. [28], which detects state changes of appliances by sensing the electromagnetic field (EMF) in the surrounding based on magnetic and electric field fluctuations. Using such cheap sensing approaches is then used by algorithms like ViridiScope [20] or Jung and Savvides' disaggregation algorithm [16]. While ViridiScope also relies on other types

[1] http://vs.inf.ethz.ch/res/show.html?what=eco-data.

[2] https://github.com/beckel/nilm-eval

of sensors (e.g., light sensors), Jung et al. assume that each appliance in the household is equipped with a binary sensor that reports whether or not an appliance is running.

3 Algorithms

Table 1 summarizes the main characteristics of the four NILM algorithms we implemented and evaluated for this study. The selected algorithms span the design space discussed in the previous section. They include supervised, unsupervised, and semi-supervised approaches as well as algorithms that require different levels of detail with respect to the measurements (i.e., real power only vs. real and reactive power). We however include only algorithms that have been developed to operate on aggregate consumption data measured at a frequency of at most 1 Hz. The reason for this restriction is that data at this granularity can be provided by most off-the-shelf electricity meters. Its collection thus does not require the costly and cumbersome installation of additional hardware. The four algorithms are briefly described in the following subsections. More details about their implementation are provided in [9].

3.1 Algorithm Description

Parson's Algorithm: The algorithm of Parson et al. [27] relies on hidden Markov models (*HMMs*) and the Viterbi algorithm [30] to disaggregate the electricity consumption of a household. For each appliance, it determines the most likely sequence of states (i.e., operating states of an appliance), depending on the observed aggregate electricity consumption, state transition probabilities, and the estimated consumption of an appliance in each state. Using this state sequence, the algorithm estimates the consumption of the appliance, subtracts it from the aggregated consumption, and then iteratively estimates the consumption of other appliances in the household. To determine the transition probabilities and power demand of each appliance, Parson et al. developed a semi-supervised training process. Instead of using submetered consumption data of an appliance, the algorithm utilizes a *generic appliance model*, which contains information on the characteristics of a certain appliance type. In case of a fridge, for instance, the algorithm incorporates information such as the average consumption of other fridges as a priori knowledge. On the basis of the generic appliance model, Parson's algorithm infers the parameters of a *specific appliance model* that describes the behavior of the appliance in the specific household.

The authors evaluated the performance of their approach on the REDD data set. They estimated the consumption of four appliances (i.e., refrigerator, microwave, clothes dryer, air conditioning) achieving a mean normalized error of 21%–77% and a root mean squared error between 77 W and 477 W using aggregated data at a granularity of 1/60 Hz.

Baranski's Algorithm: Baranski's algorithm [3] identifies recurring electricity consumption patterns in the aggregate electricity consumption and attributes those patterns to individual appliances. To this end, it extracts events (i.e., changes in electricity consumption over a given threshold) from the aggregate consumption and clusters those events, assuming that events in the same cluster belong to the same appliance. Next, a genetic algorithm creates a state machine and the most likely state sequence for each of the appliances.

Baranski's algorithm is unsupervised and thus can operate without knowing which appliances exist in the target household. The algorithm has been evaluated on both simulated data and on real-world data collected with an optical sensor in one household over a time period of about five to ten days [3]. By inspecting the resulting clusters, the authors claim to confidently identify chief consumer load devices like refrigerators, heaters, or stoves. However, although the algorithm is unsupervised, it requires to manually assign the resulting clusters or finite state machines to appliances in order to generate meaningful feedback for the occupants.

Weiss' Algorithm: Weiss' algorithm [31] extracts switching events from the household's aggregate electricity consumption and assigns each event to the appliance with the best match in a signature database. The algorithm is based on the approach developed by Hart [14], which clusters events by their real and reactive power in a training phase and assigns each event to the appliance with the best matching cluster during operation. The number of clusters is determined dynamically. In contrast to Hart, Weiss' algorithm relies on three-dimensional consumption data (i.e., real power, reactive power, and distortion power) and smoothes the power signal before extracting events. Weiss et al. also propose a novel method to unobtrusively generate signatures with the help of a smartphone application used to indicate a switching event of an appliance. Up to now, the algorithm has not been evaluated on real world consumption data. Due to the lack of a large scale signature database, we treat Weiss' algorithm as a supervised approach that is trained using plug-level data.

Kolter's Algorithm: Like Parson's algorithm, the algorithm developed by Kolter and Jaakkola [22] also models appliances as HMMs in order to disaggregate a household's electricity consumption. However, the algorithm is unsupervised as it only requires a household's aggregate electricity consumption data. To create an HMM for each appliance, the algorithm estimates the number of appliances and their consumption patterns from the aggregate consumption data. To this end, it extracts snippets of consumption data that likely correspond to an appliance's *ON cycle*, which is defined as the period between the appliance's start-up and shutdown. Next, it models each of the snippets as HMM and identifies those snippets that most likely belong to the same appliance. This results in a factorial HMM (i.e., a composition of several independent HMMs), which the authors then use to estimate the consumption of each individual appliance. To this end, they developed AFAMAP, an approximate inference technique for factorial HMMs [22]. The authors evaluated their approach on the REDD data set, analyzing 1 Hz aggregate consumption data using plug-level measurements of 7 appliances for validation. Overall, the algorithm achieved 87% precision and 60% recall.

4 Evaluation Methodology

Performing a performance evaluation of a NILM algorithm is difficult, because there is no standard evaluation procedure to apply [33]. Even more challenging is performing a fair comparison between the performance of different algorithms. This is because composition and usage of appliances

Table 1. Overview of the four NILM algorithms evaluated in this study. *Granularity* refers to the granularity of the data which the authors used to evaluate their algorithm in their original work.

Authors	Learning	Granularity	Data set	Characteristic
Parson et al. [27]	Semi-supervised	1 min (real power)	REDD [23]	Train factorial HMMs using prior knowledge of appliance types.
Baranski & Voss [3]	Unsupervised	1 sec (real power)	Simulated and real-world data (not publicly available)	Cluster switching events and apply genetic algorithm to assign events to appliances.
Weiss et al. [31]	Supervised	1 sec (real and reactive power)	Artificial lab setting	Extract switching events and find best match in signature database.
Kolter & Jaakkola [22]	Unsupervised	1 sec (real power)	REDD [23]	Generate HMMs from "snippets" identified in the aggregated consumption data.

differs significantly from time to time and from household to household. The performance of an algorithm thus highly depends on aspects such as the number of appliances running at the same time, the "noise" in the aggregated consumption data caused by unreported appliances, the performance metrics selected by the authors, and the input parameters they choose to tune their algorithm to the underlying data.

To gain a comprehensive view on the performance of a NILM algorithm it is thus necessary to run the algorithm in different scenarios (e.g., using data from different households and from multiple time periods) and to experiment with different input parameters of the algorithm. To this end, we developed a Matlab-based open source framework called NILM-Eval, which enables the evaluation of NILM algorithms on multiple data sets, households, data granularities, time periods, and specific algorithm parameters. By encapsulating those parameters in *configurations*, NILM-Eval allows the user with little effort to repeat experiments performed by others, to evaluate an algorithm on a new data set, and to fine-tune configurations to improve the performance of an algorithm in a new setting.

Figure 1 shows an overview of the framework, which we made available to the public. As input, a user provides (or selects) the implementation of an algorithm and specifies one or more *default configurations*. The default configurations provide means for the developer of the algorithm or for the user who evaluates the algorithm to adapt it to the corresponding household or data set. A user then creates *experiments* by selecting one or more default configurations and by optionally overwriting their parameters. NILM-Eval then evaluates all combinations of parameters specified by the user and thus supports evaluation a broad range of parameter combinations. For each of the combinations, NILM-Eval creates a setup file, which then serves as input for the evaluation system. Since each run is performed on a separate Matlab instance, NILM-Eval scales over many experiments (e.g., by running it on a computing cluster). Ultimately, NILM-Eval provides as results for each experiment (1) the value of each of the performance metrics supported by the algorithm, (2) the estimated consumption of each appliance or, alternatively, labeled events, and (3) a series of plots illustrating the results.

To measure the performance of a NILM algorithm, NILM-Eval supports several metrics. In case an algorithm returns the inferred electricity consumption of individual appliances, the framework computes for each appliance n the *root mean square error*, $\mathrm{RMSE} = \sqrt{\frac{1}{T}\sum_t (y_t^{(n)} - \hat{y}_t^{(n)})^2}$,

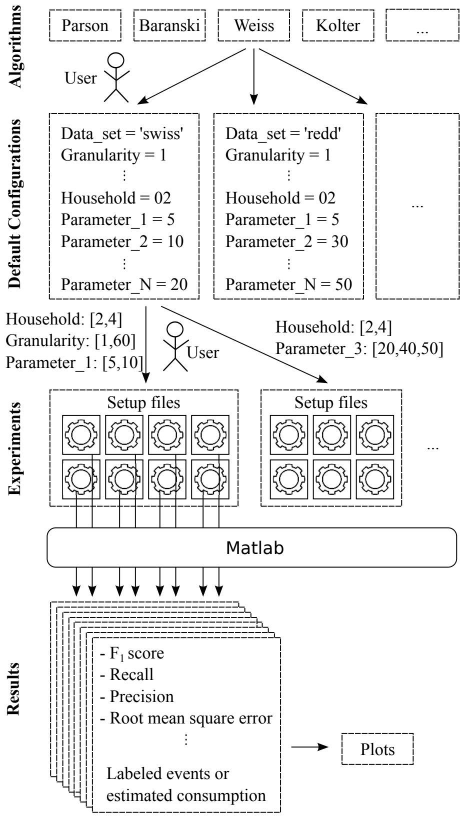


Figure 1. Evaluation framework NILM-Eval.

where $y_t^{(n)}$ denotes the actual electricity consumption of n at time t, $\hat{y}_t^{(n)}$ corresponds to n's inferred electricity consumption at time t, and T corresponds to the total number of time steps. NILM-Eval also determines the *deviation* of the inferred electricity consumption from the actual electricity consumption over a period of time, $\mathrm{Dev} = \left|\sum_{t=1}^{T} y_t^{(n)} - \sum_{t=1}^{T} \hat{y}_t^{(n)}\right| / \sum_{t=1}^{T} y_t^{(n)}$. Additionally, NILM-Eval determines the number of *true positives* (TP), *false positives* (FP), and *false negatives* (FN) for each appliance. To this end, we define an appliance-specific threshold θ. If $\hat{y}_t, y_t > \theta$ we consider $\hat{y}_t$ a true positive, if $\hat{y}_t > \theta$ and $y_t < \theta$, a false positive, and if $\hat{y}_t < \theta$ and $y_t > \theta$, a false negative. NILM-Eval then computes the *F1 score* as $\mathrm{F}_1 = 2 * \frac{\mathrm{PRC}*\mathrm{RCL}}{\mathrm{PRC}+\mathrm{RCL}}$. PRC and RCL denote the precision and recall, which are defined as $\mathrm{PRC} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \qquad \mathrm{RCL} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$.

In case the algorithm estimates switching events instead of the inferred electricity consumption at each time instant (e.g., Weiss' algorithm), NILM-Eval computes only F_1 score, precision, and recall. In this case, for an appliance n, TP corresponds to the number of events correctly assigned to n and FP to those assigned to n even though the event was not caused by n. FN denotes the number of events missed by the algorithm.

5 The ECO Data Set

The analysis presented in this paper (see section 6 below) is based on sensor data we collected from 6 households in Switzerland over a period of 8 months (June 2012 to January 2013). We refer to this data set as *ECO data set* and make it available to the research community.

In the past few years, several data sets collected for the purpose of evaluating NILM algorithms have been published. Each of those data sets exhibits different characteristics with respect to the number of households, data granularity, duration of the deployment, side information (e.g., coverage of appliances with smart plugs), and level of detail of the smart meter data (e.g., if it contains both real and reactive power). The *REDD* [23] data set, for instance, was published by Kolter and Johnson in 2011. Since then, other data sets have been published such as the *Smart** data set [4], *GREEND* [26], *BLUED* [1], *AMPds* [24], *UK-Dale* [18], *iAWE* [5], and the *Pecan Street* [15] data set.

The ECO data set extends existing data sets on four aspects. First, it contains data collected over 8 months. Only the AMPds and the UK-Dale data sets cover a comparably long time span. Second, the aggregate electricity consumption data provided with the ECO data set is very detailed as it contains measurements of real and reactive power for each of the three phases in a household. Of the other data sets, only the Smart*, the AMPds, the iAWE, and the BLUED data sets provide both real and reactive power. Third, we collected plug-level data at 1 Hz frequency, which is otherwise only provided by the Smart*, iAWE, and GREEND data sets. Last but not least, the ECO data set is to the best of our knowledge the only data set that also includes occupancy information of the households.

In [21], we already describe the characteristics of the households (e.g., number of occupants, type of household), the measurement infrastructure, and we provide details about the occupancy information of the households. In this paper, we focus on all aspects related to the plug-level measurements, which are required to evaluate the NILM algorithms. Also, we make the data set available to the public in the context of this paper. The households are named *household 1* to *household 6*. The first five households represent households 1–5 in [21], household 6 did not provide occupancy information and was thus omitted in [21].

For each of the six households we collected aggregate electricity consumption data at 1 Hz using off-the-shelf smart meters. In total we collected more than 100 million measurements during the period of the deployment. Each of the measurements contains – for each of the three phases in the household – information on voltage, current, and phase shift between voltage and current. The data can thus be used by NILM algorithms that require real and reactive power. To obtain ground truth data for our analysis we deployed 6–10 smart plugs into each of the six households. We collected measurements from each of the plugs connected to the following appliances:

- **Household 1**: (1) Fridge, (2) dryer, (3) coffee machine, (4) kettle, (5) washing machine, (6) PC, (7) freezer.
- **Household 2**: (1) Tablet, (2) dishwasher, (3) stove, (4) fridge, (5) TV, (6) stereo, (7) freezer, (8) kettle, (9) lamp, (10) laptops.
- **Household 3**: (1) Tablet, (2) freezer, (3) coffee machine, (4) PC, (5) fridge, (6) kettle, (7) entertainment.
- **Household 4**: (1) Fridge, (2) kitchen appliances[3], (3) lamp, (4) stereo & laptop, (5) freezer, (6) tablet, (7) entertainment, (8) microwave.
- **Household 5**: (1) Tablet, (2) coffee machine, (3) kettle, (4) microwave, (5) fridge, (6) entertainment, (7) PC, router & printer, (8) fountain.
- **Household 6**: (1) Lamp, (2) laptop & printer, (3) routers, (4) coffee machine, (5) entertainment, (6) fridge, (7) kettle.

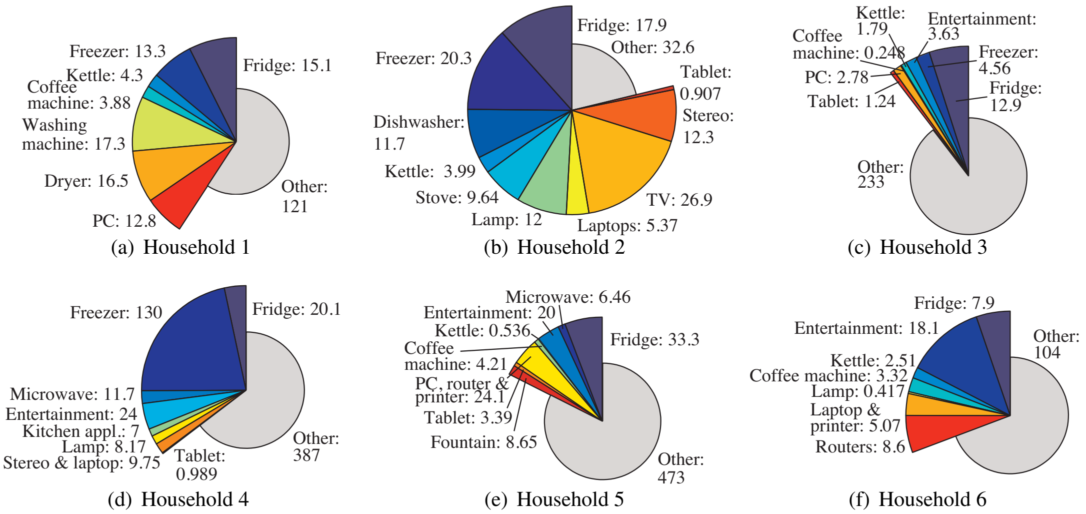
For details on the plug-level data (e.g., the number of days measured per plug), we refer to the documentation of the ECO data set[4]. In household 3, a concrete ceiling in the basement disturbed the radio connection between our gateway and the plugs. For this reason, the coverage of measurements is low for most of the appliances. We thus omit household 3 in the rest of our study and focus on the remaining five households instead.

Figure 2 shows the monthly electricity consumption of each appliance covered by the smart plugs. As we equipped only 6–10 appliances per household with a plug, each of the charts shows a significant portion named *other* that is measured by the smart meter but not attributed to any of the appliances. In household 2, roughly 80% of the electricity consumption is covered by the smart plugs. Households 3 and 5 exhibit a particularly high proportion of non-attributed consumption with more than 80%. Household 3 runs a boiler that heats water during the night. The non-attributed consumption for household 5 is high because the household uses a time-triggered pool pump, which is not covered by a plug and consumes 500 W during daytime.

In total, we collected more than 650 million measurements from 45 smart plugs deployed into the six households. As described in [21], the frequency of the plug measurements varies because we had to read them sequentially from a central gateway. To be consistent with the aggregate consumption data, we sampled the plug measurements for each appliance at 1 Hz. If a small number of values is missing between two measurements (i.e., less than 100), we replaced those missing values with the last existing measurement. If more than 100 consecutive measurements are missing, we assume that the plug has been removed and invalidated the missing values by setting them to -1.

[3] Kitchen appliances consist of a coffee machine, a bread baking machine, and a toaster.

[4] http://vs.inf.ethz.ch/res/show.html?what=eco-data



(a) Household 1 (b) Household 2 (c) Household 3

(d) Household 4 (e) Household 5 (f) Household 6

Figure 2. Electricity consumption covered by the smart plugs for each of the six households. Each of the values represents kWh per month consumed by the appliance.

We also collected occupancy information: Each of the residents manually entered presence and absence into a tablet computer during selected periods of the study, and we additionally mounted a passive infrared sensor per household next to the entrance door. Since we do not utilize the occupancy information in this work, we refer to [21] for occupancy-related information in the data set.

6 NILM Results

In this section we present the results of the performance evaluation of the four NILM algorithms described in section 3. We run the performance on the *ECO data set* and using our *NILM-Eval* framework. The results show for instance that (semi-)supervised algorithms perform better than unsupervised ones. This is mainly because the latter fail to identify consumption patterns of individual appliances in the aggregate consumption data. We further show that a data granularity of 1 Hz is required to reliably detect switching events of appliances. Weiss' algorithm, for instance, achieves F_1 scores up to 0.92 when detecting events of cooling appliances or appliances with high changes in their consumption patterns. The F_1 scores obtained by Parson's algorithm, which relies on data at 1/60 Hz frequency, are much lower and range from 0.51 to 0.80.

A fair comparison of the performance of the algorithms is however still cumbersome to obtain due to the differences on input data and training methods they rely upon as well as due to different types of output they provide. Weiss' algorithm, for instance, returns event labels, and Parson's and Kolter's algorithms return an estimate of each appliance's electricity consumption. For this reason, we present the results of the four algorithms individually and present a summary of the evaluation at the end of this section. For implementation details and parameter specifications other than the ones listed in this section, we refer to [9] and to the NILM-Eval project.

6.1 Parson's Algorithm

Like Parson et al. [27], we downsampled the data to a granularity of 1/60 Hz. We first evaluate Parson's algorithm by inferring the electricity consumption of the fridge for each of the five households, because the fridge is the only appliance that was measured by a plug in every household. Next, we evaluate the microwaves in households 4 and 5 representing appliances with switching events that change the power consumption by at least 500 W.

Fridge: We model a fridge as an appliance with two states (i.e., *ON* and *OFF* states). To define the generic model required by Parson's algorithm, we performed different initial experiments evaluating the effect of the required parameters. Ultimately, we assume the emission probabilities for each state to be Gaussian distributed and describe the *ON* state with $\mu_{on} = 60\,\mathrm{W}, \sigma^2_{on} = 40\,\mathrm{W}^2$ and the *OFF* state with $\mu_{off} = 2\,\mathrm{W}, \sigma_{off} = 5\,\mathrm{W}^2$. We further define the transition probabilities as $\phi_{on,off} = 0.2$ (i.e., the probability that the state changes from *ON* to *OFF*) and $\phi_{off,on} = 0.05$. To adapt the generic model to specific appliance models, we use 10 training days and a training window length of 3,600 seconds. The experiments are then performed five times (using different training periods) over 90 days of consumption data. Finally, to increase robustness of the Viterbi algorithm to "noise" caused by unmodeled appliances, we set the likelihood threshold introduced by Parson et al. to 0.0001.

Table 2 shows the disaggregation results we obtained by inferring the electricity consumption of the fridge from the aggregate electricity consumption of five of the households using the configuration described above. Training type *none* means that the generic model is used as input to the Viterbi algorithm. *Plug* training uses ground truth data to build a specific appliance model from the general model, whereas *aggregate* training aims at building the specific appliance model from the aggregate electricity consumption. Each value denotes the average of the five runs we performed. In terms of F_1 score, Parson's algorithm performs best (0.80)

Table 2. Performance of Parson's algorithm on the ECO data set.

Metric	Training	H.1	H.2	H.4	H.5	H.6
F_1 score	none	**0.65**	**0.64**	**0.51**	**0.52**	0.77
	plug	0.42	0.60	0.47	0.47	**0.80**
	aggregate	0.42	0.46	0.40	-	0.71
RMS	none	33 W	**37 W**	**48 W**	**62 W**	23 W
	plug	**29 W**	41 W	62 W	75 W	**17 W**
	aggregate	34 W	45 W	64 W	-	20 W
Dev	none	0.61	0.50	**0.23**	0.30	0.99
	plug	**0.31**	**0.46**	0.75	**0.27**	**0.62**
	aggregate	0.48	0.49	0.52	-	0.77

for household 6. However, this includes training on the submetered data. In real world settings (i.e., training with aggregate consumption data), the algorithm performs slightly worse with $F_1 = 0.77$. Household 6 also exhibits the lowest RMS (17 W) compared to the other households (which range from 29 W to 62 W). However, the fridge of household 6 is the most energy-efficient among all fridges in the data set, which explains that there is still a relatively large deviation of the estimation compared to the ground truth (62%).

Overall, the estimation performs better for household 6 compared to the other households. We believe this is due to the fact that households 1 to 5 have freezers, which in general have a consumption pattern that is difficult to distinguish from a fridge's consumption pattern at a 1-minute granularity. Training on aggregated data performs slightly worse than training on plug-level data. A possible explanation is that the generic model defined above is already close to the optimal model, because we performed initial experiments to carefully define the generic model.

Microwave: To evaluate the disaggregation of the microwave's electricity consumption, we rely on the default configuration provided by Parson et al.'s implementation (i.e., two states, $\mu_{on} = 1{,}700\,\text{W}$, $\sigma^2_{on} = 1{,}000\,\text{W}^2$, $\mu_{off} = 4\,\text{W}$, $\sigma^2_{off} = 100\,\text{W}^2$, $\phi_{on,off} = 0.3$, $\phi_{off,on} = 0.01$, and a likelihood threshold of 0.00001. In addition – in order to improve the performance of the estimation – we pre-processed the aggregate consumption data by replacing edges that contain more than two time steps by "sharp" edges that span exactly two time steps. With this configuration we performed five runs using aggregate training. On average, Parson's algorithm achieves F_1 scores of 0.14 and 0.031 for households 4 and 5, respectively. These low F_1 scores are caused by very low precision values, which are 0.10 and 0.017 for households 4 and 5, respectively. Parson's algorithm overestimates the consumption of the microwave as it often infers the microwave is running when it is not. By analyzing only the consumption on the phase on which the microwave is running, the F_1 scores improve to 0.18 and 0.055 for households 4 and 5, respectively.

We re-performed the experiments for both fridge and microwave, testing a variety of appliance models in order to separate the effect of training the appliance models from the actual inference. The best F_1 scores achieved by the algorithm when disaggregating the consumption of the fridge range from 0.54 (household 4) to 0.84 (household 6). In case of the microwaves in households 4 and 5, the algorithm achieved maximum values of 0.29 and 0.14, respectively.

Table 3. Event clusters in household 2 provided by Baranski's algorithm.

Cluster	ΔP	Size	App. 1	%	App. 2	%
C_1	-11 W	8,963	Laptops	28%	Fridge	3%
C_2	11 W	8,724	Laptops	31%	Fridge	3%
C_3	-58 W	3,009	Freezer	41%	Fridge	28%
C_4	73 W	1,960	Freezer	51%	Fridge	5%
C_5	93 W	1,003	Fridge	69%	Freezer	2%
C_6	-1,837 W	260	Stove	21%	Kettle	15%
C_7	1,857 W	253	Stove	21%	Kettle	14%
C_8	1,249 W	225	Stove	26%	Laptops	2%
C_9	-176 W	210	TV	5%	Freezer	4%
C_{10}	-1,235 W	199	Stove	35%	Laptops	3%
C_{11}	2,425 W	187	Stove	17%	Laptops	3%
C_{12}	-2,365 W	155	Stove	26%	Dishwasher	6%
C_{13}	-509 W	122	Freezer	14%	Fridge	9%
C_{14}	-783 W	102	Fridge	18%	Freezer	18%
C_{15}	596 W	97	Freezer	10%	Fridge	5%
C_{16}	850 W	88	Freezer	16%	Fridge	11%
C_{17}	375 W	83	Freezer	12%	Fridge	4%
C_{18}	1,064 W	60	Fridge	13%	Stove	5%
C_{19}	-1,023 W	56	Fridge	9%	Laptops	5%
C_{20}	-3,391 W	39	Stove	5%	Fridge	3%

Overall, we see the following challenges: First, consumption patterns of appliances differ considerably, which makes it difficult to define a general model that represents all appliances of a certain appliance type. Second, disaggregating each appliance in isolation leads to errors due to overlapping consumption patterns. Thus it would be interesting to apply Kolter and Jaakkola's AFAMAP algorithm [22] to infer the consumption of multiple appliances simultaneously. Finally, due to aggregation of the consumption data to 1/60 Hz, lots of details in the consumption patterns are lost. It is thus a part of our future work to evaluate Parson's algorithm using 1 Hz consumption data.

6.2 Baranski's Algorithm

We applied the unsupervised algorithm of Baranski and Voss on 30 days of aggregated 1 Hz consumption data from household 2. We set the number of resulting clusters to 20 and specified that each appliance consists of two states with a maximum length of the *ON* state set to 3,600 seconds. As a result of our initial experiments, we decided not to assign weights to the length of a switching event as well as to the boost in electricity consumption that can occur when an appliance is switched on.

Table 3 shows the clusters that result from the experiment. Each cluster denotes a set of switching events that have a similar increase (or decrease) in electricity consumption. Column *Size* shows the number of events in a cluster. By comparing the timestamps of the events with the plug-level consumption data, we assigned each event to an appliance if possible. Column % in the table shows the proportion of events assigned to the appliance named in the previous column divided by the overall number of events in the cluster. Columns *App 1.* and *App 2.* illustrate which appliances have the highest and the second highest number of assigned events in a cluster, respectively. The laptop, freezer, fridge, and stove are appliances that are often represented in the clusters. Clusters C_1 and C_2 almost exclusively contain start and stop events of the laptop. The events of the stove range across multiple clusters (i.e., clusters C_6, C_7, C_8, C_{10}, C_{11}, and C_{12}), because the change in electricity consumption

Table 4. Finite state machines provided as a result by Baranski's algorithm.

FSM	$\Delta P(C_1)$	$\Delta P(C_2)$	Sequences	Duration	App.
1	1,857 W	-1,837 W	276	115 s	Stove or Kettle
2	1,249 W	-1,235 W	312	14 s	Stove
3	11 W	-11 W	12,066	14 s	Laptops
4	2,425 W	-2,365 W	260	10 s	Stove
5	1,064 W	-1,023 W	56	63 s	?
6	850 W	-783 W	144	12 s	Freezer
8	596 W	-509 W	138	17 s	Freezer
10	1,249 W	-1,023 W	30	78 s	Stove
11	73 W	-58 W	3,032	627 s	Freezer

of the stove events varies. Note that clusters C_6 and C_7 are also populated to a large extent by switching events of the kettle, which makes forming a state machine for the stove and for the kettle difficult. The events of the fridge and the freezer are also spread over multiple clusters.

Based on the clustering results, Baranski's algorithm generated the finite state machines (FSMs) shown in table 4. The second and third columns denote the power steps of the centroids of the two clusters that form the FSM. The next two columns list the number of sequences represented by the FSM as well as their average duration. Column *App.* denotes the appliance that is most likely represented by the FSM. Note that this labeling has been performed manually. The first FSM consists of events from clusters 1 and 2 and therefore represents the stove or the kettle. The third FSM represents the laptop, which is the only appliance that is clearly separable from the other appliances. FSMs 2, 4, and 10 also represent the stove, whereas FSMs 6, 8, and 11 represent the freezer. In case of the freezer, the first two FSMs exhibit high consumption and last only shortly, which is why we assume they are caused by the initial spike in the consumption pattern at the beginning of a cooling cycle. The fridge is not represented in the list of FSMs. The reason is that, although the ON event of the fridge is well represented in cluster C_5, the corresponding OFF events are spread over multiple clusters. Baranski's algorithm computed a quality score for each FSM and thus discarded the FSM(s) that represent(s) the fridge due to a low quality score.

In practice, Baranski's algorithm requires the user to manually label the resulting FSMs without the assignments of events to appliances as provided in table 3. Even so, the algorithm generates multiple FSMs for some of the appliances due to the fact that some of the clusters contain events from multiple appliances. Therefore, there is a large ambiguity in the assignment of appliances to the FSMs. Possible improvements include (a) to allow creating an FSM using events from different clusters to reduce the number of FSMs, and (b) to improve the clustering procedure. Possible improvements of the clustering include using real and reactive power to make events of different appliances more distinguishable, or to apply post-processing that divides or combines clusters (e.g., on the basis of the number of events in each cluster).

6.3 Weiss' Algorithm

We use the 1 Hz consumption data of household 2 including real and reactive power split into individual phases to evaluate Weiss' algorithm. We investigate appliances of the three categories (1) cooling appliances, (2) appliances with high consumption (i.e., dishwasher, kettle, stove), and (3) remaining appliances (i.e., lamp, laptop, TV, stereo system). The analysis is based on 90 days of consumption data plus 15 days of data used for training. In the training process, we extract timestamps of switching events (i.e., changes in power consumption above 5 W) from the plug data and extract the signature from the smart meter data at these timestamps.

Table 5. Signatures of cooling appliances (top), appliances with high consumption (center), and other appliances (bottom) in household 2.

Appliance	Event	Δ Real	Δ Reactive	Phase
Fridge	OFF	-69.2 W	-5.9 VA	1
Fridge	ON	79.9 W	4.4 VA	1
Freezer	OFF	-51.6 W	16.5 VA	1
Freezer	ON	63.8 W	-20.0 VA	1
Dishwasher	OFF	-2,058 W	3.8 VA	1
Dishwasher	ON	2,060 W	-18.3 VA	1
Kettle	OFF	-1,881 W	2.4 VA	1
Kettle	ON	1,853 W	-4.3 VA	1
Kettle	ON	1,884 W	3.8 VA	1
Stove	OFF	-903 W	-519 VA	1&2
Stove	ON	626 W	315 VA	1&2
Lamp	OFF	-185 W	-111 VA	1
Lamp	OFF	-185 W	-216 VA	1
Lamp	ON	222 W	91.4 VA	1
Lamp	ON	127 W	87.2 VA	1
Laptops	OFF	-20.2 W	-3.4 VA	1
Laptops	ON	23.2 W	10.3 VA	1
TV	OFF	-166 W	-35.7 VA	2
TV	ON	159 W	30.1 VA	2
TV	ON	161 W	32.6 VA	2
Stereo	OFF	-17.3 W	-11.8 VA	2
Stereo	ON	55.6 W	48.8 VA	2

Table 6. Performance results achieved by Weiss' algorithm on consumption data from household 2.

	F_1 score	Precision	Recall	TP	FP	FN
Fridge	0.92	0.93	0.91	4,855	385	477
Freezer	0.92	0.98	0.86	5,948	137	947
Dishwasher	0.56	0.95	0.39	115	6	178
Kettle	0.75	0.95	0.62	122	6	74
Stove	0.24	1.0	0.14	28	0	209
Lamp	0.30	0.37	0.25	23	39	68
Laptops	0.11	0.10	0.12	63	593	498
TV	0.37	0.89	0.24	90	11	291
Stereo	0.10	0.23	0.06	144	471	2,148

Table 5 shows the signatures of household 2's appliances extracted from the aggregate consumption data. For each appliance, the table shows the change in real power (Δ Real) and reactive power (Δ Reactive) at ON or OFF switching events. Column *Phase* illustrates on which phase the appliance is running. For cooling cycles, a switching event denotes the beginning or the end of the cooling cycle. For appliances with switching events of more than 500 W real power, only those events are considered during training and recognition phase. For the other appliances, each event with more than 5 W is considered a switching event. The table shows that the events of the fridge and of the freezer have differences in both real power (16 W to 18 W difference on average) and reactive power (22 VA to 24 VA), which is a good property in order to be distinguished by Weiss' algorithm. The stove runs both on phase 1 and on phase 2, the TV and stereo system run on phase 2, and the other appliances run on phase 1.

Table 6 illustrates the results achieved by Weiss's algorithm for each of the appliances. Identifying the switching events of the fridge and the freezer is possible with F_1 scores of 0.92 each. In case of the freezer, the algorithm misses only 196 out of 5,992 switching events, which is a precision of 0.98. Events from appliances with high consumption, namely dishwasher, kettle, and stove, are recognized with almost no false positives, leading to a precision of 0.95, 0.95, and 1.0, respectively. However, the algorithm misses a large number of events for these appliances. This is why the F_1 scores are 0.56, 0.75, and 0.25, respectively. The remaining appliances exhibit relatively low F_1 scores. In case of the lamp, this is due to the fact that household 2 has a dimmable lamp, which means that the power steps caused by switching events vary. The laptop and the stereo system are difficult to reliably identify, because their power consumption is very low and can be easily confused with switching events or variations caused by other appliances.

Overall, Weiss' algorithm performs well for cooling appliances and for appliances with high power consumption. In the latter case, the precision is very high, but the algorithm misses many events and thus exhibits low recall values. The algorithm includes a scaling parameter r to control the maximum distance of a switching event to the (possibly) corresponding signature. Increasing r leads to a higher recall value, because the algorithm identifies more switching events of a particular appliance. However, this step also results in a higher number of false positives. To reduce the number of false positives, we recommend including additional features such as time of day or the relationship between certain appliances (e.g., the dryer often runs after the washing machine).

6.4 Kolter's Algorithm

As described in section 3, Kolter's algorithm automatically identifies and clusters snippets (i.e., consumption patterns of appliances) before it disaggregates the consumption data. Using the data from household 2, we analyzed 7 days of 1 Hz real power consumption data searching for three types of snippets (i.e., snippets with one, two, and three *ON* states).

The algorithm detected 399 snippets with one *ON* state, 221 snippets with two *ON* states, and 136 snippets with three *ON* states. Most of the snippets with one *ON* state have a mean power consumption between 0 W and 200 W. Whereas we can attribute many of those to the freezer, the number of snippets we can attribute to the fridge (i.e., snippets with mean power consumption between 60 W and 85 W) is relatively low. The reason is that the frequency of the freezer's cooling cycles is almost twice as high as the frequency of the fridge's cooling cycles. Thus almost all cooling cycles of the fridge interfere with the cooling cycles of the freezer. There are also 44 snippets above 200 W (i.e., snippets with a mean power consumption of 1,200 W, 1,800 W, and 2,400 W). Most of these snippets represent the stove. Thus the stove and the freezer are the only appliances with a single *ON* state for which we can reliably identify snippets in the consumption data using Kolter's algorithm.

Table 7. Centroids of clusters of the snippets with two *ON* states. P_1 and P_2 denote the mean power consumption of the two *ON* states of the snippets.

Cluster	P_1	P_2	Snippets
C_1	1263 W	21 W	2
C_2	320 W	733 W	3
C_3	65 W	42 W	23
C_4	7 W	52 W	12
C_5	1278 W	1278 W	2

For the snippets with two or three *ON* states, we applied k-means clustering to obtain cluster centroids that potentially represent the consumption pattern of individual appliances. Table 7 shows the 5 resulting cluster centroids for the snippets with two *ON* states. We compared all cluster centroids with the consumption patterns measured by the smart plugs, but we could not find a match between any of the cluster centroids and the consumption pattern of an appliance. The same holds for the snippets with three *ON* events. Since the application of the spectral clustering method did not result in HMMs that represent individual appliances, we decided to omit Kolter and Jaakkola's second step. Instead, we propose to extract snippets using the plug-level data in order to evaluate Kolter and Jaakkola's AFAMAP algorithm.

6.5 Summary of the Results

The results show that supervision is required to achieve reasonable performance. Weiss' algorithm and Parson's algorithm perform better than the unsupervised approaches investigated in our this study, which do not reliably identify appliances in the aggregate consumption data. One reason for this is the lack of periods in which only a single appliance is running. In household 2, for instance, 98.4% of the fridge's cooling cycles are overlapped by a cooling cycle of the freezer. As an additional constraint, each of the unsupervised approaches requires manual labeling after recognizing the appliances. For these reasons, Parson's semi-supervised approach is promising as it assumes generic appliance models and fine-tunes them given the aggregate consumption data. We further observe that Weiss' algorithm performs better than Parson's algorithm. We believe this is due to the fact that Weiss's algorithm utilizes fine-grained consumption data (i.e., measured at 1 Hz on multiple phases including real and reactive power), whereas Parson's algorithm uses data sampled over a period of 1 minute. Using Weiss' algorithm, we can reliably identify events from cooling appliances as well as from appliances with high electricity consumption such as the stove or the dishwasher.

In addition to optimizing these four algorithms as proposed in each of the subsections, we see potential in combining the algorithms. Using Parson et al.'s generic appliance model to generate HMM's followed by Kolter and Jaakkola's AFAMAP algorithm, for instance, combines the advantages of both approaches. Leveraging the strength of Weiss et al.'s algorithm in recognizing switching events could further provide valuable input to the HMM-based approaches by providing information on appliance state changes.

7 Limitations and Future Work

A limitation of each NILM evaluation – as with many data-driven approaches – is that the results highly depend on the data used and on the configuration of the algorithm parameters. For this reason, we collected consumption data over a particular long time frame and developed our evaluation system NILM-Eval to test a variety of combinations of parameters for each of the algorithms. We still observe a high variance in the results depending on the configuration of the household. It is therefore a part of our future work to extend the analysis to a larger number of households and compare the stability of the results on different data sets.

We evaluated each of the algorithm using the same data granularity and learning method than the authors did in their original evaluation. However, the performance found in our analysis is likely insufficient for real world applications. For this reason, we will investigate the performance of the algorithms under different requirements in our future work. For instance, we plan on applying Parson's algorithm on 1 Hz consumption data, and aim at training Kolter's algorithm on plug-level data rather than identifying the number and type of appliances in an unsupervised way.

Relaxing the requirements to the input data, we plan to evaluate algorithms that utilize context information provided by other sources [7, 16, 20]. ON/OFF state transitions of appliances, for instance, can be reported by smart appliances themselves or by other sensors such as electromagnetic field sensors [28]. Another example consists in the inclusion of occupancy information, which can be sensed by a smartphone and is also provided as a part of the ECO data set.

8 Conclusions

In this paper we present both a comprehensive data set and an evaluation framework to analyze the performance of NILM algorithms and demonstrate the use of the framework on four selected NILM approaches. Our results show that thanks to the use of our framework the suitability of selected approaches to be used in real scenarios as well as their limitations can be assessed. Both the presented data set and the evaluation framework are made publicly available.

9 Acknowledgments

We would like to thank the participating households from Energie Thun for their collaboration. We further thank Friedemann Mattern for his valuable comments and Daniel Pauli for his support collecting the smart meter data. This work has been supported by the Hans L. Merkle Foundation.

10 References

[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 *Proc. SustKDD*, 2012.

[2] K. C. Armel, A. Gupta, G. Shrimali, and A. Albert. Is disaggregation the holy grail of energy efficiency? The case of electricity. *Energy Policy*, 52:213–234, 2012.

[3] M. Baranski and J. Voss. Genetic algorithm for pattern detection in NIALM systems. In *Proc. SMCS*. IEEE, 2004.

[4] 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. In *Proc. SustKDD*. ACM, 2012.

[5] N. Batra, M. Gulati, A. Singh, and M. B. Srivastava. It's different: Insights into home energy consumption in India. In *Proc. BuildSys*. ACM, 2013.

[6] 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 *Proc. e-Energy*. ACM, 2014.

[7] C. Beckel, W. Kleiminger, T. Staake, and S. Santini. Improving device-level electricity consumption breakdowns in private households using ON/OFF events. *ACM SIGBED Review*, 9(3), 2012.

[8] M. Berges, E. Goldman, H. S. Matthews, L. Soibelman, and K. Anderson. User-centered nonintrusive electricity load monitoring for residential buildings. *Journal of Computing in Civil Engineering*, 25(6):471–480, 2011.

[9] R. Cicchetti. NILM-Eval: Disaggregation of real-world electricity consumption data. Master's thesis, ETH Zurich, 2014.

[10] S. Darby. The effectiveness of feedback on energy consumption. A review for DEFRA of the literature on metering, billing and direct displays. 2006.

[11] L. Farinaccio and R. Zmeureanu. Using a pattern recognition approach to disaggregate the total electricity consumption in a house into the major end-uses. *Energy and Buildings*, 30(3):245–259, 1999.

[12] C. Fischer. Feedback on household electricity consumption: A tool for saving energy? *Energy Efficiency*, 1(1):79–104, 2008.

[13] S. Gupta, M. S. Reynolds, and S. N. Patel. ElectriSense: Single-point sensing using EMI for electrical event detection and classification in the home. In *Proc. UbiComp*. ACM, 2010.

[14] G. W. Hart. Nonintrusive appliance load monitoring. *Proc. of the IEEE*, 80(12):1870–1891, 1992.

[15] C. Holcomb. Pecan Street Inc.: A test-bed for NILM. In *Proc. NILM Workshop*, 2012.

[16] D. Jung and A. Savvides. Estimating building consumption breakdowns using ON/OFF state sensing and incremental sub-meter deployment. In *Proc. SenSys*. ACM, 2010.

[17] J. Kelly and W. Knottenbelt. Metadata for energy disaggregation. In *Proc. CDS*. IEEE, 2014.

[18] J. Kelly and W. Knottenbelt. 'UK-DALE': A dataset recording UK domestic appliance-level electricity demand and whole-house demand. *ArXiv e-prints*, 2014. arXiv:1404.0284.

[19] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han. Unsupervised disaggregation of low frequency power measurements. In *Proc. SDM*. SIAM, 2010.

[20] Y. Kim, T. Schmid, Z. Charbiwala, and M. Srivastava. ViridiScope: Design and implementation of a fine grained power monitoring system for homes. In *Proc. UbiComp*. ACM, 2009.

[21] W. Kleiminger, C. Beckel, T. Staake, and S. Santini. Occupancy detection from electricity consumption data. In *Proc. BuildSys*. ACM, 2013.

[22] J. Z. Kolter and T. Jaakkola. Approximate inference in additive factorial HMMs with application to energy disaggregation. In *Proc. AISTATS*, 2012.

[23] J. Z. Kolter and M. J. Johnson. REDD: A public data set for energy disaggregation research. In *Proc. SustKDD*, 2011.

[24] S. Makonin, F. Popowich, L. Bartram, B. Gill, and I. V. Bajic. AMPds: A public dataset for load disaggregation and eco-feedback research. In *Proc. EPEC*. IEEE, 2013.

[25] A. Marchiori, D. Hakkarinen, Q. Han, and L. Earle. Circuit-level load monitoring for household energy management. *IEEE Pervasive Computing*, 10(1):40–48, 2011.

[26] A. Monacchi, D. Egarter, W. Elmenreich, S. D'Alessandro, and A. M. Tonello. GREEND: An energy consumption dataset of households in Italy and Austria. In *Proc. SmartGridComm*. IEEE, 2014.

[27] O. Parson, S. Ghosh, M. Weal, and A. Rogers. Nonintrusive load monitoring using prior models of general appliance types. In *Proc. AAAI*, 2012.

[28] A. Rowe, M. Berges, and R. Rajkumar. Contactless sensing of appliance state transitions through variations in electromagnetic fields. In *Proc. BuildSys*. ACM, 2010.

[29] S. Spiegel and S. Albayrak. Energy disaggregation meets heating control. In *Proc. SAC*. ACM, 2014.

[30] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. on Information Theory*, 13(2):260–269, 1967.

[31] M. Weiss, A. Helfenstein, F. Mattern, and T. Staake. Leveraging smart meter data to recognize home appliances. In *Proc. PerCom*. IEEE, 2012.

[32] M. Zeifman and K. Roth. Nonintrusive appliance load monitoring: Review and outlook. *IEEE Trans. on Consumer Electronics*, 57(1):76–84, 2011.

[33] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar. Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey. *Sensors*, 12(12):16838–16866, 2012.