NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring

Nipun Batra[1], Jack Kelly[2], Oliver Parson[3], Haimonti Dutta[4], William Knottenbelt[2], Alex Rogers[3],Amarjeet Singh[1], Mani Srivastava[5]

[1]Indraprastha Institute of Information Technology, Delhi, India {nipunb, amarjeet}@iiitd.ac.in
[2] Imperial College, London {jack.kelly, w.knottenbelt}@imperial.ac.uk
[3] University of Southampton {op106, acr}@ecs.soton.ac.uk
[4] CCLS, Columbia {haimonti@ccls.columbia.edu}
[5] UCLA {mbs@ucla.edu}

1. ABSTRACT

Non-intrusive load monitoring, or energy disaggregation, aims to separate household energy consumption data collected from a single point of measurement into individual appliances. In recent years, the field has rapidly expanded due to increased interest as national deployments of smart meters have begun in many countries. However, empirically comparing such algorithms is currently virtually impossible, as a result of the different data sets used, the lack of reference implementations of these algorithms, and the variety of accuracy metrics employed. To address this challenge, we present the Non-intrusive Load Monitoring Toolkit (NILMTK); an open source toolkit designed specifically to enable the comparison of energy disaggregation algorithms in a reproducible manner. Our toolkit includes parsers for a range of existing data sets, a standard output format for disaggregation algorithms, a set of statistics for describing data sets, two reference benchmark disaggregation algorithms, and a suite of accuracy metrics. We demonstrate that our toolkit makes it easier to enter into energy disaggregation research by simplifying the use of multiple data sets, while supporting the addition of new disaggregation algorithms, and also encouraging direct comparisons to be made between algorithms through common output formats and accuracy metrics. In summary, this work is the first research to compare multiple disaggregation approaches across multiple publicly available data sets.

2. INTRODUCTION

Non-intrusive load monitoring (NILM), or energy disaggregation, aims to break down a household's aggregate electricity consumption into individual appliances [**?**]. The motivations for such a process are threefold. First, informing a household's occupants of how much energy each appliance consumes empowers them to take steps towards reducing their energy consumption [**?**]. Second, personalised feedback can be provided which quantifies the savings of certain appliance-specific advice, such as the financial savings when an old inefficient appliance is replaced by a new efficient appliance. Third, if the NILM system is able to determine the time of use of each appliance, a recommender system would be able to inform the household's occupants of the potential savings through deferring appliance use to a time of day when electricity is either cheaper or has a lower carbon footprint.

Such benefits have drawn significant interest in the field since its inception 25 years ago. In recent years, the combination of smart meter meter deployments [**?**, **?**] and also decreasing hardware costs of household electricity sensors has lead to a rapid expansion of the field. Such rapid growth over the past 5 years has been evidenced by the wealth of academic papers published, international meetings held (e.g. NILM 2012, EPRI NILM 2013), startup companies founded (e.g. Bidgely, Neurio) and data sets released (e.g. REDD, BLUED, Smart*).

However, three core obstacles currently prevent the direct comparison of state-of-the-art approaches, and as a result are limiting progress within the field. First, each contribution to date has only been evaluated on a single data set and consequently it is hard to assess the generality of any proposed approach. Furthermore, many papers sub-sample data sets to select specific households, appliances and time periods, making experimental results more difficult to reproduce. Second, newly proposed approaches are rarely compared against the same benchmark algorithms, further increasing the difficulty in empirical comparisons of performance between different publications. Moreover, the lack of reference implementations of these state-of-the-art algorithms often leads to the reimplementation of such approaches. Third, each paper targets a different use case for NILM and therefore evaluates the accuracy of their proposed approach using a different set of

performance metrics. As a result the numerical performance calculated by such metrics cannot be compared between any two papers. These three barriers obstacles have led to the proposal of successive extensions to state-of-the-art algorithms, while a direct comparison between such new and existing approaches has remained impossible.

Similar obstacles have arisen in other research fields and prompted the development of toolkits specifically designed to support research in that area. For example, PhysioToolkit offers access to over 50 databases of physiological data and provides software to support the processing and analysis of such data for the biomedical research community [**?**]. Similarly, CRAWDAD collects 89 data sets of wireless network data in addition to supporting software to aid the analysis of such data by the wireless network community [**?**]. However, no such toolkit is available to the NILM community.

Against this background, we propose NILMTK[1]; an open source toolkit designed specifically to enable the comparison of energy disaggregation algorithms. The primary aim of the toolkit is to provide a complete pipeline from data sets to accuracy metrics, therefore lowering the barrier to entry for researchers to plug in a new algorithm and compare its performance against the current state of the art. The toolkit has been released as open source software in an effort to encourage researchers to contribute data sets, benchmark algorithms and accuracy metrics as they are proposed, with the goal of enabling a greater level of collaboration within the community. In addition, the toolkit has been designed using a modular structure, therefore allowing researchers to reuse or replace individual components as required. Last, the toolkit has been written in Python with flat file input and output formats, in addition to high performance binary formats, ensuring compatibility with existing algorithms written in any language and designed for any platform.

Our contributions are summarised as follows:

- We propose the NILMTK-DF (data format), the standard energy disaggregation data structure used by NILMTK. NILMTK-DF is modelled loosely on the REDD data set format [**?**]. Furthermore, we provide parsers from multiple existing data sets into our proposed NILMTK-DF format. In addition, we propose a single output format for the disaggregated data estimated by NILM algorithms.
- We provide a set of statistics functions for analysing and identifying issues with each data set. We also provide a set of preprocessing functions for mitigating common challenges with NILM data sets.
- We provide implementations of two benchmark disaggregation algorithms: first an approach based on combinatorial optimisation [**?**], and second an approach based on the factorial hidden Markov model [**?**, **?**]. We demonstrate the ease by which NILMTK allows the comparison of these algorithms across a range of existing data sets, and present results of their performance.
- We present a suite of accuracy metrics which are able to evaluate the performance of any disaggregation algorithm which produces an output compatible with NILMTK. This allows the performance of a disaggregation algorithm to be evaluated for a range of use cases.

The remainder of this paper is organised as follows. In Section **??** we provide an overview of related work from the field of NILM and also other similar fields of research. In Section **??** we present NILMTK, and give a detailed description of its components. In Section **??** we demonstrate the empirical evaluations which are enabled by NILMTK, and provide analysis of existing data sets and accuracy metrics. Finally, in Section **??** we conclude the paper and propose directions for future work.

3. BACKGROUND

The field of non-intrusive load monitoring was founded over 20 years ago when Hart proposed the algorithmic disaggregation of household energy usage [**?**]. Since then, many studies have shown the benefits of disaggregated data to both household occupants through financial savings, and also to utility providers through load shedding potential [**?**, **?**]. However, the majority of research had been evaluated using either lab-based data or simulated data, hence the performance of disaggregation algorithms in real households had remained unknown. More recently, national deployments of smart meters have prompted a renewed interest in energy disaggregation, and as a result a number of data sets collected specifically for energy disaggregation have been released. We now discuss the data sets which are currently available, which we subsequently summarise in Table **??**.

In 2011, the Reference Energy Disaggregation Dataset (REDD) [**?**] was the first data set to be released which was collected specifically to aid NILM research. The data set contains both aggregate and sub-metered power data from 6 homes, and has since become the most popular data set for evaluating energy disaggregation algorithms. In 2012, the Building-Level fUlly-labeled dataset for Electricity Disaggregation (BLUED) [**?**] was released containing data from a single household. However, the data set does not include sub-metered power data, and instead events were recorded each time an appliance changed state (e.g. turned on). As a result, it is only possible to evaluate how well appliance activity can

[1] `github.com/nilmtk/nilmtk`

Data set	Institution	Location	Duration per house	Number of houses	Appliance sample frequency	Aggregate sample frequency
REDD (2011)	MIT	MA, USA	3-19 days	6	3 secs	1 sec and 15 kHz
BLUED (2012)	CMU	PA, USA	8 days	1	N/A	N/A
Smart* (2012)	UMass	MA, USA	3 months	1	1 sec	1 sec
Tracebase (2012)	Darmstadt	Germany	N/A	N/A	1-10 second	N/A
Sample (2013)	Pecan Street	TX, USA	7 days	10	1 min	N/A
HES (2013)	DECC, DEFRA	UK	1 or 12 months	251	2 or 10 minute	2 or 10 minute
AMPds (2013)	Simon Fraser U.	BC, Canada	1 year	1	1 min	Yes
iAWE (2013)	IIIT Delhi	Delhi, India	73 days	1	1 sec	1 sec
UKPD (2014)	Imperial College	London, UK	3-14 months	4	6 secs	1-6 secs or 16 kHz

Table 1: Table of comparison of household energy data sets.

be inferred, rather than the disaggregation of appliance energy consumption. More recently, the SMART* [**?**] data set was released, which contains household aggregate power data from 3 homes, while sub-metered appliance power data was only collected from a single household.

In 2013 the Pecan Street sample data set was released [**?**], which contains both aggregate data and sub-metered data from 10 households. Later, the Household Energy Study data set was released, which contains data from 251 households although only sub-metered appliance data was collected. Also that year, the Almanac of Minutely Power dataset (AMPds) [**?**] was released which contains both aggregate and sub-metered data from a single household. Subsequently, the Indian data for Ambient Water and Electricity Sensing (iAWE) [**?**] was released, which contains both aggregate and sub-metered electricity data from a single home. Most recently, the UK Power Dataset (UKPD) [**?**] was released which contains data from four households using both aggregate meters and individual appliance sub-meters. Unfortunately, subtle differences in the aims of each data set have lead to completely different data formats being used, and as a result a time-consuming engineering barrier exists when moving from one data set to another. This has resulted in publications using only a single data set to evaluate a given approach, and consequently the generality of results are rarely investigated. Having described the data sets that are currently available, we now discuss recent publications which have used such data sets for the evaluation of disaggregation algorithms.

The REDD data set has been used to compare the performance of many disaggregation algorithms since its publication in 2011. The data set was proposed along with a performance result of a benchmark disaggregation algorithm using 10 second data across 5 of the 6 households [**?**]. Kolter and Jaakkola later proposed an extension to the benchmark algorithm [**?**], however the extension was only evaluated using features extracted from 14 kHz data from a single home from the data set, and therefore the performance results are not directly comparable. Later, Zeifman [**?**] and Johnson and Willsky [**?**] evaluated various approaches using the same data set, although both selected a different subset of appliances and calculated an artificial household aggregate from these appliances, therefore simplifying the disaggregation problem and preventing a numerical comparison with other publications. Subsequently, Parson et al. [**?**] and Rahayu et al. [**?**] both proposed new approaches, although each were evaluated using a different set of 4 houses from the REDD data set, again preventing a numerical comparison between publications. Last, Batra et al. [**?**] evaluated their approach on the REDD data set using a different household to Kolter and Jaakkola, again preventing a direct performance comparison of numerical results. As a result, it has not been possible to deduce whether one approach is preferable to another from the literature.

In addition to REDD, other data sets have been used to evaluate the performance of NILM algorithms. For example, BLUED was introduced along with a benchmark algorithm [**?**], but has since only been used by one other publication [**?**]. Similarly, AMPds and iAWE have only been used to evaluate disaggregation algorithms proposed by the data set authors [**?**, **?**]. Clearly, the variety of different formats is slowing the uptake of new data sets, and also preventing algorithms from being tested across multiple data sets. Having discussed the open data sets used by recent publications, we now describe the benchmarks against which newly proposed disaggregation algorithms have been compared.

It is essential to compare newly proposed disaggregation algorithms to the state of the art in order to assess the increase in an algorithm's performance. However, the lack of available reference implementations of state-of-the-art disaggregation algorithms has led to authors often comparing against more basic benchmark algorithms. This problem is further compounded since there is no single consensus on which benchmarks to use, and

as a result most publications use a different benchmark algorithm. For example, Kolter and Jaakkola compared their approach to a set of decoupled HMMs [**?**], Parson et al. and Batra et al. both evaluated their approaches against variants of their own approaches [**?**, **?**], Zeifman compared their approach to a Bayesian classifier, while Rahayu et al. and Johnson and Willsky both compared against a FHMM [**?**, **?**]. Clearly, further publications would benefit from openly available common benchmark algorithms against which newly proposed algorithms could be easily compared. Having discussed common benchmark algorithms, we now describe the various metrics which have been used to compare disaggregation approaches.

The range of different application areas of energy disaggregation has prompted a number of evaluation metrics to be proposed. For example, four disaggregation metrics labelled *energy correctly assigned* have recently been used to evaluate the performance of disaggregation algorithms using the REDD data set. First, Kolter and Johnson [**?**] proposed an accuracy metric which captures the error in assigned energy normalised by the actual energy consumption in each time slice averaged over all appliances, which was also later used by Rahayu et al. [**?**] and Johnson and Willsky [**?**]. However, large errors in the assigned energy in some time slices will result in a negative accuracy, making this an ill-posed metric. Second, Kolter and Jaakkola [**?**] proposed an equivalent metric, although the error is presented individually for each appliance rather than an average across all appliances. Third, Parson et al. [**?**] proposed a similar metric, although the metric captures the error in assigned energy consumed over the complete duration of the data set rather than per time slice. However, this metric allows overestimates and underestimates in the assigned energy in different time slices to cancel out, and therefore does not represent all disaggregation errors. Fourth, Batra et al. [**?**] proposed a subtly different metric to Kolter and Johnson [**?**], in which error is reported instead of accuracy, and also energy assigned to an incorrect appliance is double counted. Each of the differences between these four metrics prevents a numerical comparison between publications, and motivates the use of common metrics. Having described the various metrics which have been used by recent publications, we now discuss the suitability of general purpose machine learning toolkits for energy disaggregation.

Although no toolkit currently exists specifically for energy disaggregation, various toolkits are available for more general machine learning tasks. For example, scikit-learn is a general purpose machine learning toolkit implemented in python [**?**], while GraphLab is machine learning and data mining toolkit written in C++ [**?**]. While such toolkits provide implementations of generic machine learning algorithms, they lack functionality specific to the energy disaggregation domain, such as data set parsers, benchmark disaggregation algorithms, and energy disaggregation metrics. Therefore, an energy disaggregation toolkit should extend such general toolkits rather than replace them, in a similar way that scikit-learn adds machine learning functionality to the numpy numerical library for Python. This further motivates a toolkit designed specifically for energy disaggregation research.

4. PROBLEM DEFINITION

The aim of energy disaggregation is to provide estimates, $\hat{y}_t^{(n)}$, of the actual power demand, $y_t^{(n)}$, of each appliance n at time t, from household aggregate power readings, $\bar{y}_t$. Most NILM algorithms model appliances using a set of discrete states such as off, on, intermediate, etc. We use $x_t^{(n)} \in \{1, \ldots, K\}$ to represent the ground truth state, and $\hat{x}_t^{(n)}$ to represent the appliance state estimated by a disaggregation algorithm.

5. NILMTK

NILMTK aims to address the following three design goals:

1. **Analysis**: The toolkit should facilitate the easy analysis of NILM data sets and expose the entire pipeline from data import through to disaggregation metrics.

2. **Ease of adding new algorithms**: The toolkit should provide consistent interfaces for new disaggregation algorithms.

3. **Ease of deployment**: The toolkit should be built in such a way that learnt household models can be easily deployed and are able to interface with online as well as offline data.

We implemented NILMTK in Python due to the availability of a vast set of libraries supporting both machine learning research (e.g. Pandas, Scikit-learn) and the deployment of such research as web applications (Django). Furthermore, Python allows easy deployment in diverse environments including academic settings and is increasingly being used for data science. Our API design is heavily based on scikit-learn [**?**, **?**], which is a well known machine learning framework in the Python community. Scikit-learn is designed for general purpose machine learning tasks and has set high code and documentation standards. However, as discussed earlier in Section **??**, the domain specifics of NILM demand a specialised framework.

Figure **??** presents the NILMTK pipeline from the import of data sets to the evaluation of various disaggregation algorithms over NILM specific metrics. We discuss each module of the NILMTK pipeline below.

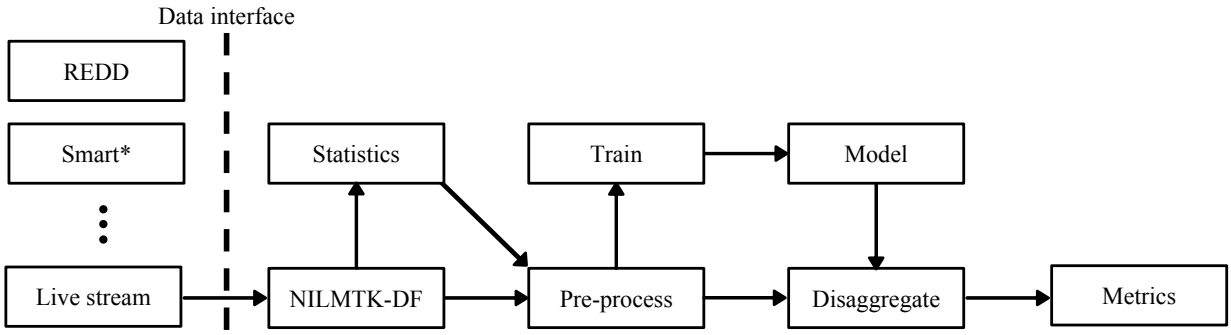

Figure 1: NILMTK pipeline. At each stage of the pipeline, results and data can be stored into or loaded from the disk.For deployment: You may also give an input of real time data here and give one output as real time disaggregation.

5.1 Data Format

As discussed in Section **??**, a number of data sets have been released recently. However, differences between these data sets have resulted in low adoption, limiting the ease by which NILM algorithms can be compared. Motivated by this, we propose NILMTK-DF, a common data set format inspired by the REDD format [**?**], into which existing data sets can be converted. NILMTK currently includes importers for the following data sets: REDD, PECAN, HES, iAWE, AMPds and UKPD. However, it was not possible to include an importer for BLUED due to the lack of sub-metered appliance power data and also Tracebase was not included due to the lack of information regarding which appliances were present in the same household.

After import, the data resides in our NILMTK-DF in-memory data structure, which is used throughout the NILMTK pipeline. Data can be saved or loaded from disk at multiple stages in the NILMTK processing pipeline to allow other tools to interact with NILMTK. We provide two CSV flat file formats: a rich NILMTK-DF CSV format and a "strict REDD" format which allows researchers to use their existing tools designed to process REDD data. We also provide a more efficient binary format using the Hierarchical Data Format (HDF5). In addition to storing electricity data, NILMTK-DF can also store relevant metadata and other sensor modalities such as gas, water, temperature, etc. It has been shown in the past that such additional sensor and metadata information may help enhance NILM prediction [**?**].

Another important feature of our format is naming standardisation. Different data sets use different labels for the same class of appliance (e.g. REDD uses 'refrigerator' whilst AMPds uses 'FGE') and different names for the measured parameters. When data is first imported into NILMTK, these diverse labels are converted to a standard vocabulary.

In addition, NILMTK allows rich metadata to be associated with a building, appliance or meter. For example, NILMTK can store the parameters measured by each meter (e.g. reactive power, real power), the geographical coordinates of each building (useful for exploring the relationship between appliance usage and weather), the mains wiring defining the meter hierarchy (useful if a single appliance is measured at the appliance, circuit and aggregate levels), whether a single meter measures multiple appliances, which appliances are not sub-metered and also whether a specific light is dimmable. A full description of NILMTK-DF is provided in the appendix.

Through such a combination of metadata and naming standardisation, NILMTK allows data analysis to be performed over multiple data sets. For example, users can perform queries such as: "what is the average energy consumption of refrigerators in the USA compared to the UK?"

5.2 Data Set Statistics and Diagnostics

Due to a variety of challenges in collecting real world data, no data set is free of errors. Hence, researchers often need to spend a considerable amount of time exploring the detailed characteristics of each data set before proceeding with their research. One such common problem arises due to missing data. Short gaps could be the result of the loss of a small number of radio transmissions from a wireless sensor. However, larger gaps are more likely to be caused by either a faulty or switched off sensor. To help diagnose these issues, NILMTK provides the following four functions: a function to calculate the proportion of samples which are missing across an entire data set, a function to plot the dropout rate over time, a function to get the total sensor up-time and a function to get the boundaries of complete sections of data.

NILMTK also provides functions for exploring appliance usage. For example, the toolkit can calculate

the distribution of appliance usage over repeating time periods, correlations between appliances and weather or other appliances, and the distributions of appliance power demands, on-durations and off-durations. Furthermore, to explore the proportion of aggregate data which has been sufficiently sub-metered, NILMTK provides functions to calculate the proportion of time slices where a minimum percentage of the total energy has been sub-metered and also produces a list of the top-k appliances across the time slices.

5.3 Preprocessing

The data sets described above have been collected with a range of hardware, with different aims and under different settings. As shown in Table **??**, the sampling frequency of appliance monitors varies from 1 second to 2 minutes across the data sets. Thus, we provide filters to down-sample data sets at a specified frequency. Further, these data sets have been collected from different countries, where the voltage fluctuations vary widely. Batra et al. showed voltage fluctuations from 180-250 V in the iAWE data set collected in India [**?**], while the voltage variation in the SMART* data set is in the range 118-123 V. Hart et al. showed the need to factor these voltage fluctuations as they can significantly impact power draw [**?**]. Therefore, we also add a voltage normalisation function defined as follows:

$$Power_{normalised} = \left(\frac{Voltage_{nominal}}{Voltage_{observed}}\right)^2 \times Power_{observed} \quad (1)$$

As we explain in the algorithms section below, the memory required by some NILM algorithms is exponential in the number of appliances. Thus, when high number of appliances are considered, the models do not fit into memory and distributed algorithms may be needed. This necessitates the need to filter out top-k appliances by their contribution which can be in terms of energy or power. We also created preprocessing functions for fixing other common issues with these datasets, such as 1) filling small periods of missing data when appliance sensors did not report readings; 2) filtering out implausible values (such as when readings where observed voltage is more than twice the rated voltage); 3) filtering out appliance data when mains data is missing, etc.

5.4 Training and Disaggregation

NILMTK provides implementations of two common benchmark NILM algorithms: combinatorial optimisation (CO) and factorial hidden Markov model (FHMM). CO was proposed by Hart in his seminal work [**?**], while techniques based on extensions of the FHMMs constitute the state-of-the-art. The aim of the inclusion of these algorithms is not to present state-of-the-art disaggregation results, but instead to enable new approaches to be compared to well studied benchmark algorithms without requiring the reimplementation of such algorithms. We now briefly describe these two algorithms.

Combinatorial Optimisation: CO finds the *optimal* combination of appliance states, which minimises the difference between the sum of the predicted appliance power and the observed aggregate power.

$$\hat{x}_t^{(n)} = \underset{\hat{x}_t^{(n)}}{\operatorname{argmin}} \left| \bar{y}_t - \sum_{n=1}^{N} \hat{y}_t^{(n)} \right| \quad (2)$$

Since each time slice is considered as a separate optimisation problem, each time slice is assumed to be independent. CO resembles the subset sum problem and thus is NP-complete. The complexity of disaggregation for T time slices is $O(TK^N)$, where N is the number of appliances and K is the number of states for each appliance. Since the complexity of CO is exponential in the number of appliances, the approach is only computationally tractable when a small number of appliances are modelled.

Factorial Hidden Markov Model: Some equation or a state diagram will be useful. REDD paper has such a diagram- not sure if drawing the same one adds much value!The power demand of each appliance can be modelled as the observed value of a Gaussian hidden Markov model (HMM). The hidden component of these HMMs are the states of the appliances. Energy disaggregation involves jointly decoding the power draw of n appliances and hence a factorial HMM [**?**] is well suited. A FHMM can be represented by an equivalent HMM in which each state corresponds to a different combination of states of each appliance. The complexity of disaggregation for such a model is $O(TK^{2N})$, and as a result FHMMs scale even more poorly than CO. Such a FHMM model has three parameters: i) prior probability (π) containing K^N entries, ii) transition matrix (A) containing $K^N * K^N$ or K^{2N} entries, and iii) emission matrix (B) containing $2K^N$ entries. From an implementation perspective, even storing (or computing) A for 14 appliances with 2 states each needs 8 GB of RAM. Hence, we propose to validate FHMMs on preprocessed data where either top-k appliances are considered, or appliances contributing less than a threshold are discarded.

For certain algorithms like FHMMs, modeling relationship amongst consecutive samples is necessary. Thus, NILMTK provides facilities for dividing data into train and test while still maintaining the notion of time.

5.5 Model Import and Export

Recently, a lot of interest has arisen in deploying live NILM systems, as evidenced by the many startup companies (e.g. Energy Aware, PlotWatt, Bidgely, etc.) currently aiming to provide disaggregated consumption feedback to end consumers. However, a major barrier which currently exists is that academic algorithms are only

suitable for offline statical analysis of fixed data sets. Motivated by reducing the barrier from offline algorithms to those suitable for live deployments, we developed the Model module in NILMTK which encapsulates the results of the training module required by the disaggregation module. It is the responsibility of the model developer to create export and import functions to interface with a JSON file. NILMTK currently includes importers and exporters for both the FHMM and CO approaches. Further, in the use case section, we show how if the rated power is known, we can avoid learning on sub-metered data. At the time of deployment, this learnt JSON model can be easily imported and disaggregation performed.

5.6 Accuracy Metrics

As discussed in Section **??**, a range of accuracy metrics are required due to the diversity of application areas of energy disaggregation approaches. To satisfy this requirement, NILMTK provides a set of standard metrics to enable the evaluation of disaggregation algorithms. This will allow researchers to evaluate different energy disaggregation algorithms using identical accuracy measures. We now give a brief description of each metric along with its mathematical definition.

Error in total energy assigned: Represents the difference between the total energy assigned to appliance n and the actual energy consumed by appliance n over the data set.

$$\left| \sum_t y_t^{(n)} - \sum_t \hat{y}_t^{(n)} \right| \tag{3}$$

Fraction of total energy assigned correctly: Represents the overlap between the fraction of total energy assigned to each appliance and the actual fraction of total energy consumed by each appliance over the data set.

$$\sum_n min \left(\frac{\sum_n y}{\sum_{n,t} y}, \frac{\sum_n \hat{y}}{\sum_{n,t} \hat{y}} \right) \tag{4}$$

Normalised error in assigned power: Represents the sum of the differences between the power assigned to appliance n and the actual power of appliance n in each time slice t, normalised by the total energy consumption of appliance n.

$$\frac{\sum_t \left| y_t^{(n)} - \hat{y}_t^{(n)} \right|}{\sum_t y_t^{(n)}} \tag{5}$$

RMS error in assigned power: Represents the root mean squared error between the power assigned to appliance n and the actual power of appliance n in each time slice t.

$$\sqrt{\frac{1}{T} \sum_t (y_t - \hat{y}_t)^2} \tag{6}$$

Confusion matrix: Represents the number of time slices in which each of an appliance's states were either confused with each other state or correctly classified.

True positives, False positives, False negatives, True negatives: Represents the number of time slices in which appliance n was either correctly classified as being on (TP), classified as being on while it was actually off (FP), classified as off while is was actually on (FN), and correctly classified as being off (TN).

$$TP^{(n)} = \sum_t and \left(x_t^{(n)} = on, \hat{x}_t^{(n)} = on \right) \tag{7}$$

$$FP^{(n)} = \sum_t and \left(x_t^{(n)} = off, \hat{x}_t^{(n)} = on \right) \tag{8}$$

$$FN^{(n)} = \sum_t and \left(x_t^{(n)} = on, \hat{x}_t^{(n)} = off \right) \tag{9}$$

$$TN^{(n)} = \sum_t and \left(x_t^{(n)} = off, \hat{x}_t^{(n)} = off \right) \tag{10}$$

True/False positive rate: Represents the fraction of time slices in which appliance n was correctly predicted to be on that it was actually on (TPR), and the fraction of time slices in which appliance n was incorrectly predicted to be on that it was actually off (FPR).

$$TPR^{(n)} = \frac{TP}{(TP + FN)} \tag{11}$$

$$FPR^{(n)} = \frac{FP}{(FP + TN)} \tag{12}$$

Precision, Recall: Represents the fraction of time slices in which appliance n was correctly predicted to be on that it was actually off (Precision), and the fraction of time slices in which appliance n was correctly predicted to be on that it was actually on (Recall)

$$Precision^{(n)} = \frac{TP}{(TP + FP)} \tag{13}$$

$$Recall^{(n)} = \frac{TP}{(TP + FN)} \tag{14}$$

F-score: Represents the harmonic mean between precision and recall.

$$F\text{-}score^{(n)} = \frac{2.Precision.Recall}{Precision + Recall} \tag{15}$$

Hamming loss: Represents the total information lost when appliances are incorrectly classified over the data set.

$$HammingLoss = \frac{1}{T} \sum_t \frac{1}{N} \sum_n xor \left(x_t^{(n)}, \hat{x}_t^{(n)} \right) \tag{16}$$

In APPENDIX we summarise the NILMTK pipeline with a code snippet to illustrate the ease by which data

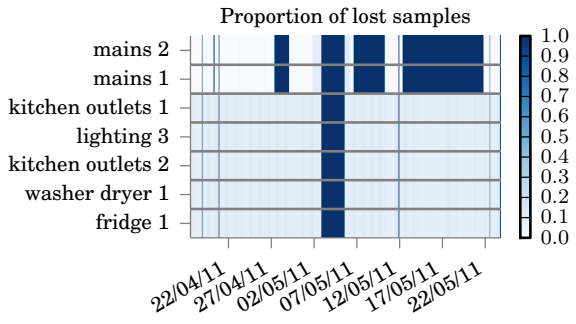

Figure 2: Lost samples from REDD house 1. Light blue indicates a low dropout rate; dark blue indicates a high dropout rate. The figure only shows a subset of the submetered channels.

sets can be imported and preprocessed, algorithms can be trained and used to disaggregate a household's energy usage, and accuracy metrics can be employed to evaluate their accuracy.

6. EVALUATION

Having introduced NILMTK in the previous section, we now demonstrate the type of analysis supported by the toolkit. First, we show how existing data sets can be processed to produce representations of the quality of data and the also to provide detail regarding the appliances present in the data set. Second, we give some examples of the effect of voltage normalisation on the power demand of individual appliances, and discuss how this might affect the performance of a disaggregation algorithm. Third, we present summary performance results of the two benchmark algorithms included in NILMTK across four data sets using a number of accuracy metrics. Fourth, we present detailed results of these algorithms for a single data set, and discuss each algorithm's performance for different appliances.

6.1 Summary of datasets

Figure **??** shows a selection of statistics across multiple public datasets. These statistics were calculated by the stats functions in NILMTK.

Figure **??** shows the distribution of lost samples for REDD house 1. From this we can see that the two mains channels have four large gaps but if we ignore these large gaps then the mains channels have a dropout rate near to zero. The submetered channels have only one large gap but, ignoring this gap, the submetered channels have a constant dropout rate of about 20%.

6.1.1 Appliance power profiles

Each appliance consumes a range of powers. Figure **??** displays histograms of the distribution of powers used by a selection of appliances. Appliances like toasters and kettles tend to have only two possible power states: on and off. This simplicity makes them amenable to be modelled by, for example, Markov chains with only two states per chain. Appliances like washing machines, hoovers, dimmable lights and computers often have more than two power states.

The proportion of energy use per appliance varies from country to country. For example, the building recorded in India for the iAWE dataset has two air conditioning units; whilst none of the buildings in UKPD have an air conditioning unit. The differences in appliance energy use by country are illustrated in figure **??**.

6.1.2 Usage patterns which could be learnt by a disaggregation system

Modern automatic speech-to-text systems not only learn to map phonemes to characters but also learn which phonemes tend to appear together and which words tend to appear together. In a similar fashion, a disaggregation system might learn not just to recognise the appearance of individual appliances in an aggregate signal but might also learn when particular appliances are usually active each day (e.g. the TV is usually on in the evening), correlations between appliances (e.g. the office LCD screen is on when the office PC is on) and correlations between appliances and additional information such as weather data from the local meteorological office (e.g. less sunshine means more usage of electric lights).

One probabilistic learning framework capable of capturing this information is the Conditional Factorial Hidden Markov Model (CFHMM) proposed by [**?**], and those same authors also presented several examples of such patterns. Prior knowledge from public datasets could be used to create general probability distributions for each appliance class and these distributions could then be refined by the disaggregation system for each house.

In the following sections, we present patterns of appliance usage per day and per week; correlations between appliance usage and weather variables; and histograms of appliance on-durations.

6.1.3 Usage histograms

Histograms showing usage patterns for a selection of appliances over an average day are shown in figure **??**. Strong similarities between groups of appliances can be seen. For example, the usage pattern of the TV and "amp livingroom" are very similar because the amp is used to play the TV's audio but the amp is sometimes on without the TV. Similarly, the kitchen lights, kettle and toaster show similar usage patterns.

To produce these usage histograms, we applied a manually-configured power threshold to each appliance. We were

Data set	Number of appliances	Percentage energy sub-metered	Percentage missing samples (ignoring gaps)	Mains up-time per building (days)	Percentage up-time
REDD	9, 16, 23	58, 71, 89	0, 10, 16	4, 18, 19	8, 40, 79
Pecan Street	13, 14, 22	75, 87, 150	0, 0, 0	7, 7, 7	100, 100, 100
AMPds	20	97	0	364	100
iAWE	10	48	8	47	93
UKPD	4, 12, 49	19, 48, 82	0, 7, 22	36, 102, 404	73, 84, 100

Table 2: Table of data set results calculated by the statistics functions in NILMTK. Each cell represents the range of values across all buildings per data set and per statistic. The three numbers per cell are the minimum, median and maximum values. AMPds and iAWE each contain just a single building, hence these rows have a single number per cell. Large gaps in each channel were removed prior to calculating "percentage energy sub-metered", "percentage missing samples" and "mains up-time". (By "gap" we mean any period between two samples longer than 20 × the sample period). iAWE data was cropped to 2013/6/11 to 2013/7/31. todo: investigate which chans are causing two Pecan homes to have % energy submetered above 100%, do Smart*

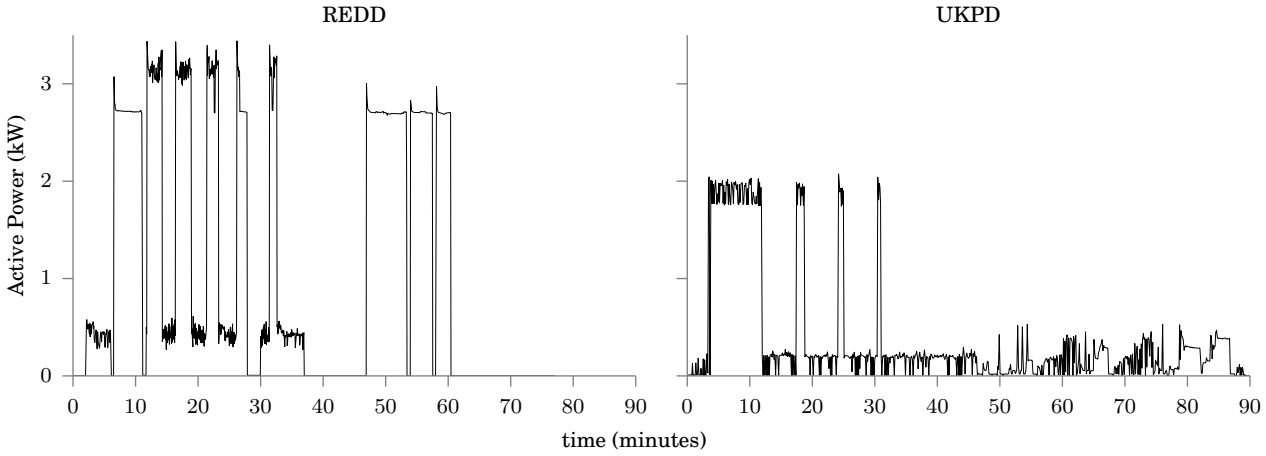

Figure 3: Comparing American and UK washing machines.

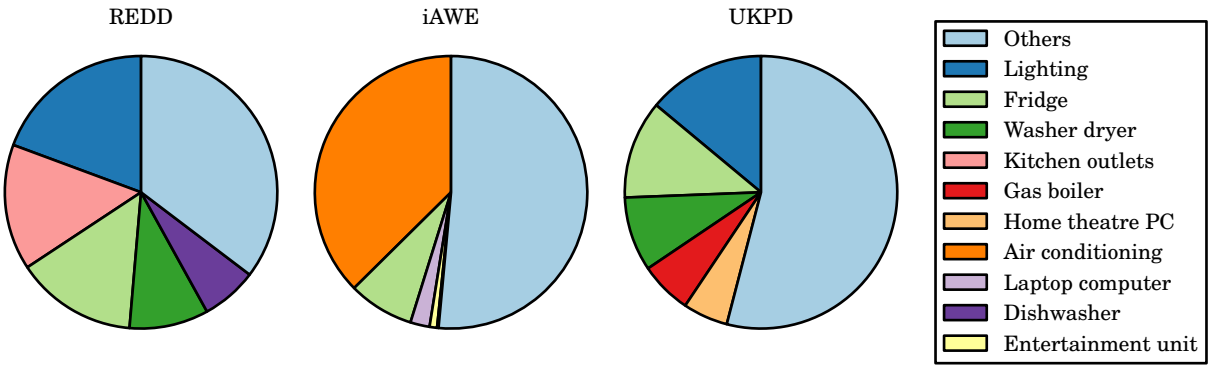

Figure 4: Top 5 appliances in terms of the proportion of the total energy used in building 1 in each of REDD (USA), iAWE (India) and UKPD (UK). We summed the energy use of appliances of the same type.

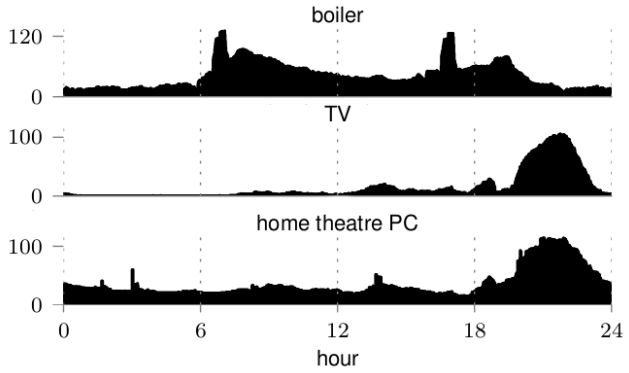

Figure 5: Daily appliance usage histograms for a selection of appliances. We used a bin width of one minute. Data from UKPD building 1.

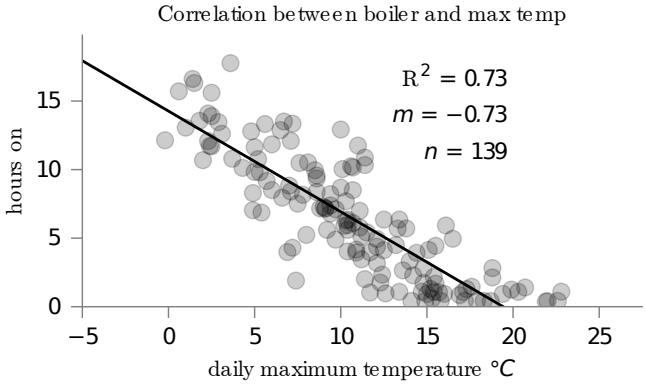

Figure 6: Linear regression showing correlation between gas boiler usage and external temperature. R^2 denotes the coefficient of determination, m is the gradient of the regression line and n is the number of data-points used in the regression. Each data-point represents one day. Appliance data from UKPD building 1. Historical daily averages from Heathrow weather station (20 miles west of UKPD building 1) were obtained from the UK Met Office under their Educational program. Days when the appliance usage was zero were ignored because we assume that the house was unoccupied on these days.

interested in user behaviour but the data set was recorded using UTC timestamps and the data set spans a daylight saving transition from GMT to BST. As such, all times were converted to local time to produce these histograms. The use of local time has produced one artefact: the small "bump" towards the right of the solar thermal pump histogram in figure **??** (the earth's rotation does not alter in response to day light saving transitions!).

6.1.4 Correlations with weather

We expect the usage of some appliances to be correlated with some weather variables. Previous studies have demonstrated correlations between temperature and heating/cooling demand in Australia[**?**] and between temperature and total household electricity usage in America [**?**]. We wanted to search for similar correlations.

If robust correlations can be demonstrated then a disaggregation system could learn correlations between weather variables and appliance usage in order to refine its appliance usage estimates. cite Kolter's December 2013 paper Weather data are relatively easy to acquire programmatically: for example, the UK Metoffice provides free access to live weather data via their DataPoint API.

Figure **??** shows correlations between boiler usage and maximum temperature. The correlation between external maximum temperature and boiler usage is strong ($R^2 = 0.73$); and it is noteworthy that the X-axis intercept ($\approx 19\,^{\circ}\mathrm{C}$) is approximately the set point for the boiler thermostat, as one might expect. Lighting circuit usage also correlates with solar radiation (not shown) although there is considerable variation in lighting usage which is not explained by solar radiation.

6.2 Voltage normalisation

Mains voltage is not constant. In the UK, for example, it is nominally 230 V but is allowed to vary by -6%, $+10\%$. This variation in voltage can produce a variation in power consumption of -12% to +20% in linear loads like resistive heaters because $P = I \times V$ and because current changes with voltage for resistive loads according to Ohm's law ($I = {}^{V}/_{R}$). These abrupt changes in reported power due to voltage fluctuations can be problematic for disaggregation algorithms. The mitigate this effect, we can normalise the power consumption, as outlined in section **??**.

Figure **??** shows histograms for both the normalised and un-normalised appliance power consumption. Normalisation produces a noticeably tighter power distribution for linear resistive devices such as the toaster but has little effect on constant power appliances such as TVs whose power consumption does not vary significantly as voltage varies because their power supplies are designed to draw the same power across the permissible voltage range.

6.3 Disaggregation across datasets

We now compare the disaggregation results across the first home of each of the following X datasets: AMPds, PECAN, iAWE, Smart*, UKPD and REDD. Since all the datasets were collected for different durations, we used fist half the samples for training and remaining half for testing across all the datasets. Further, we preprocessed datasets such as REDD, UKPD and iAWE to

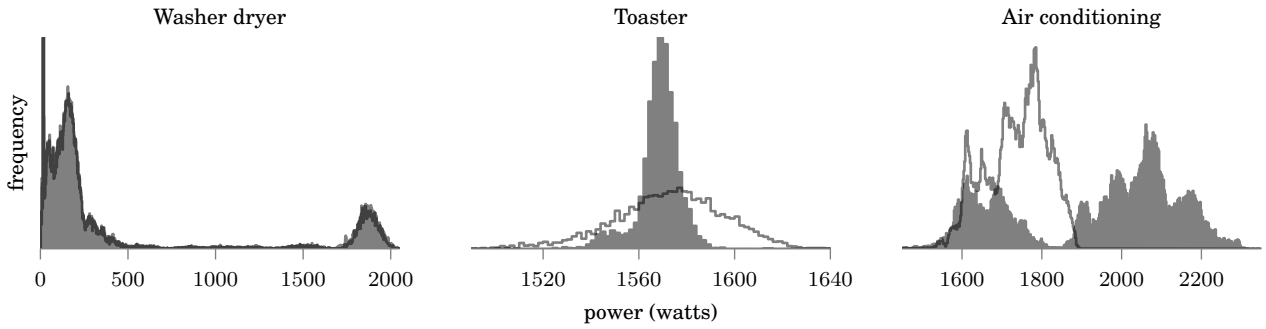

Figure 7: Histograms of power consumption. The filled grey plots show histograms of "normalised power". The thin, grey, semi-transparent lines drawn over the filled plots show histograms of unnormalised power. The "air conditioning" data is from iAWE, the "washer dryer" and "toaster" are from UKPD building 1.

1 minute frequency using the downsampling filter to account for different aggregate and mains data sampling frequencies and compensating for intermittent lost data packets. As highlighted above, REDD and iAWE also have significant proportion of missing data, which were appropriately preprocessed. AMPds and PECAN data did not require any preprocessing. Apart from REDD, Smart*, UKPD and iAWE, the remaining datasets were processed at the frequency at which they were made available (i.e. PECAN - 1 minute, AMPds - 1 minute). Since both CO and FHMM have exponential computational complexity in the number of appliances, we consider only those appliances whose contribution was greater than 5%. Across all the datasets, the appliances which contribute more than 5% of the aggregate include HVAC appliances such as the air conditioner and thermostats, and appliances which are used throughout the day such as the refrigerator. We compare the disaggregation performance of CO and FHMM across the following three metrics: 1) Fraction of total energy assigned correctly (FTE), 2) Normalised error in assigned power (NEP), 3) F-score. These three metrics and their variations have been used most often in previous work. A lower NEP indicates better performance, while in contrast a higher F-score and FTE indicate better disaggregation accuracy. The evaluation was performed on a laptop having 2.3 GHz i7 processor and 8 GB RAM running Linux.

Table **??** summarises the results of the two algorithms across the four data sets. It can be observed that FHMM performance is superior to CO performance across the three metrics for both REDD and AMPds. This confirms the theoretical foundations proposed by Hart [**?**]; that CO is highly sensitive to small variations in the aggregate load. FHMM approach overcomes these shortcoming by considering an associated transition probability between the different states of an appliance. However, it can be seen that CO performance is similar to FHMM performance in iAWE, PECAN, Smart* and UKPD across all metrics. This is likely due to the fact that in used homes from these datasets, very few appliances contribute more than 5% of the aggregate. For instance, space heating contributes very significantly (about 60% for a single air conditioner which has a power draw of 2700 W in the PECAN home and about 35% across two air conditioner having a power draw of 1800W and 1600W respectively). As a result, these appliances are easier to disaggregate by both algorithms, owing to their relatively high power demand in comparison to appliances such as electronics and lighting. Whereas, in the UKPD home, washing machine was one of the appliance contributing more than 5% of the aggregate, which brought down overall metrics across both approaches.

Another important aspect to consider is the time required for testing and training. These timings confirm the fact that CO is exponentially quicker than FHMM. This raises an interesting insight: In homes such as the ones used from PECAN and iAWE in the above analysis, it may be beneficial to use CO over a FHMM owing to the less amount of time required for training and testing, even though FHMMs are in general considered to be more powerful.

6.4 Detailed disaggregation results for a single dataset

Having compared disaggregation results across different datasets, we now do a detailed discussion of disaggregation results across different appliances for a single home in the iAWE dataset in Table **??**. As before, we model all appliances using 2 states. In order to show the disagregation performance across appliances, we choose the top 6 appliances by their energy consumption. We observe that CO and FHMM report similar performance

Dataset	Train time (s)		Disaggregate time (s)		NEP		FTE		F-score	
	CO	FHMM	CO	FHMM	CO	FHMM	CO	FHMM	CO	FHMM
REDD	3.67	22.81	0.14	1.21	1.61	1.35	0.77	0.83	0.31	0.31
SMART*	1.51	1.66	0.03	0.09	0.75	0.70	0.89	0.93	0.80	0.79
PECAN	1.72	2.83	0.02	0.12	0.68	0.75	0.99	0.87	0.77	0.77
AMPds	5.92	298.49	3.08	22.58	2.23	0.96	0.44	0.84	0.55	0.71
iAWE	1.68	8.90	0.07	0.38	0.91	0.91	0.89	0.89	0.73	0.73
UKPD	1.06	11.42	0.10	0.52	3.66	3.67	0.81	0.80	0.38	0.38

Table 3: Comparison of CO and FHMM across multiple datasets

Appliance	NEP		F-score	
	CO	FHMM	CO	FHMM
Air conditioner 1	0.3	0.3	0.9	0.9
Air conditioner 2	1.0	1.0	0.7	0.7
Entertainment unit	4.2	4.1	0.3	0.3
Fridge	0.5	0.5	0.8	0.8
Laptop computer	1.7	1.8	0.3	0.2
Washing machine	130.1	125.1	0.0	0.0

Table 4: Comparison of CO and FHMM across different appliances in iAWE dataset

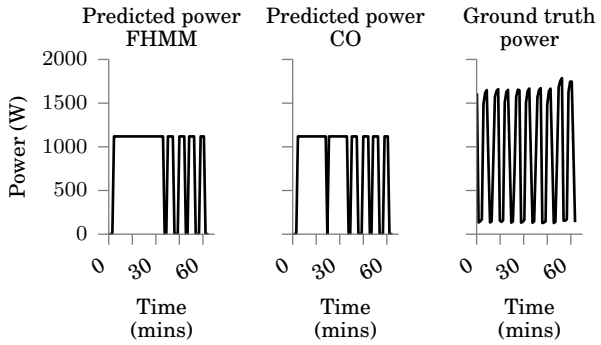

Figure 8: Comparison of predicted power (CO and FHMM) with ground truth for air conditioner 2 in the iAWE dataset

across all the appliances. We observe that across both metrics results for appliances such as washing machine and SMPS based appliances such as laptop and entertainment unit(television) are much worse when compared to HVAC loads like air conditioners. It is well known in prior literature that appliances such as washing machines are hard to model [**?**].

We observe that the performance accuracy of air conditioner 2 is much worse than air conditioner 1. This is due to the fact that during the instrumentation, air conditioner 2 was operated at a set temperature of 26 °C. With external temperature roughly $5 - 10$ °C below this set temperature, this air conditioner reached the set temperature quickly and turned off the compressor while still running the fan. However, air conditioner 1 was operated at operated at 16 °C and mostly had the compressor on. Thus, air conditioner 2 spent much more time in this intermediate state (compressor off, fan on) in comparison to air conditioner 1. Figure **??** shows how both FHMM and CO are able to detect on and off events of air conditioner 2. Since air conditioner 2 spent considerable amount of time in the intermediate state, the learnt 2 state model is less appropriate in comparison to the 2 state model used for air conditioner 1. This can be further seen in Figure **??**, where we observe that both FHMM and CO learn a much lesser power level of around 1100 W, in comparison to the rated power of around 1600 W. We believe that this can be corrected by choosing a more appropriate 3 state model for this air conditioner, which comes at a cost of increased training and disaggregation computational and memory requirements. The air conditioner set temperature details quoted above were provided by the iAWE authors as a part of appliance metadata in their dataset release.

7. USE CASE STUDY

We now consider two deployment scenarios where NILMTK is being used to perform disaggregation. Firstly, we consider the faculty home deployment at IIIT Delhi [**?**]. 20 faculty homes have been instrumented with smart meters. Data from these homes is being aggregated into a sMAP [**?**] server instance residing inside IIIT Delhi. Aggregate home data from sMAP is pulled via HTTP

request and is fed into NILMTK via a converter. The aim of this experiment was to understand if rated power can suffice for disaggregating large HVAC based loads such as air conditioner and room heaters. Thus, we fed the rated power of different appliances into CO JSON model and disaggregated on the aggregate data. We found that this approach worked sufficiently well in disaggregating HVAC based loads. The second user study was a single home deployment in Delhi/London where the data from the smart meter was collected using a low power RPi/Intel Atom.

8. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed NILMTK; an open source toolkit designed to allow empirical comparisons to be made between existing energy disaggregation algorithms. The toolkit defines a common data format, NILMTK-DF, provides parsers from X publicly available data sets to NILMTK-DF, and provides data set statistics and preprocessing functions to identify and mitigate common problems with NILM data sets. In addition, the toolkit includes implementations of two benchmark disaggregation algorithms based on combinatorial optimisation and the factorial hidden Markov model. Furthermore, NILMTK includes implementations of a set of performance metrics which will enable future research to directly compare disaggregation approaches through a common set of accuracy measures.

Paragraph on benefits of benchmark algo evaluations.

Future work will focus upon the addition of recently proposed disaggregation algorithms. For instance, both benchmark implementations included in NILMTK are well studied algorithms which require sub-metered appliance data from each household for a supervised training phase, while algorithms proposed in more recent research [**?**, **?**] only require aggregate data for an unsupervised training phase. These algorithms are computationally expensive so a GPU implementation may be explored.

An additional direction for future work would be the inclusion of a household simulator within NILMTK. Since all data sets represent a limited number of households, and it is currently impossible to test how a disaggregation algorithm might perform in a household other than those present in existing data sets. However, a simulator would overcome this problem by generating data for new households by either combining appliance data from multiple households or simulating appliances using detailed appliance models.

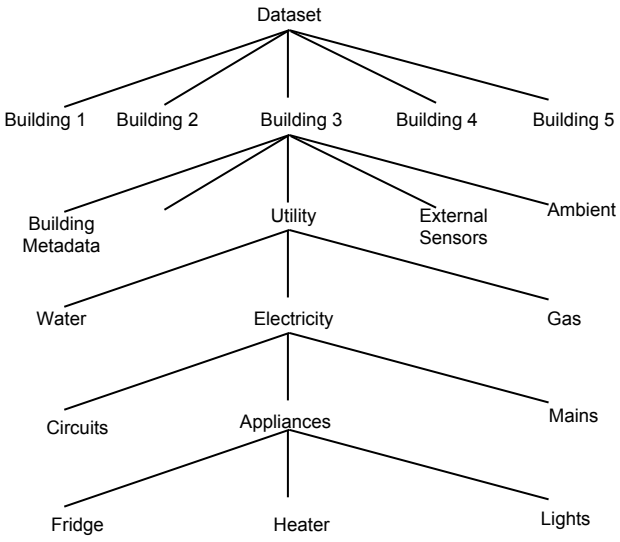

Figure 9: NILMTK-DF format hierarchy

APPENDIX

A. NILMTK-DF

We now provide the details of NILMTK-DF. NILMTK-DF follows a hierarchical structure which models the physical hierarchies.

B. SAMPLE CODE FOR NILMTK PIPELINE

We now illustrate the NILMTK pipeline via a minimal code example.

```
dataset = DataSet()

#Load the dataset
dataset.load_hdf5(DATASET_PATH)

#Load first building
building = dataset.buildings[1]

#Remove records where voltage>260 or voltage<160
building = filter_out_implausible_values(
    building, Measurement('voltage', ''), 160, 260)

#Downsample to 1 minute
building = downsample(building, rule='1T')

# Choosing feature for disaggregation
DISAGG_FEATURE = Measurement('power', 'active')

# Dividing the data into train and test
train, test = train_test_split(building)

# Train on DISAGG_FEATURES using FHMM
disaggregator = FHMM()
disaggregator.train(train,
```

```
            disagg_features=[DISAGG_FEATURE])

# Disaggregate
disaggregator.disaggregate(test)

# F1 score metric
f1_score = f1(disaggregator.predictions,
            test)
```

C. ADDING A NEW NILM ALGORITHM

Every algorithm needs to define the following four functions:

- train: The parameters of this function are the 'building', The parameters are inspired from R style formulas.
- disaggregate
- import model:
- export model: