Chapter 3

Non-intrusive load monitoring toolkit (NILMTK)

3.1 Introduction

While NILM is an old field, spanning more than three decades of research, three core obstacles prevented the direct comparison of state-of-the-art approaches, and as a result impeded progress within the field. To the best of our knowledge, each contribution to date had only been evaluated on a single data set and consequently it is hard to assess whether such approaches generalise to new households. Furthermore, many researchers sub-sampled data sets to select specific households, appliances and time periods, making experimental results more difficult to reproduce. Second, newly proposed approaches were 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 led to the reimplementation of such approaches. Third, many papers targeted different use cases for NILM and therefore the accuracy of their proposed approaches are evaluated 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 obstacles have led to the proposal of successive extensions to state-of-the-art algorithms, while a direct comparison between new and

existing approaches remains 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 [42]. Similarly, CRAWDAD collects 89 data sets of wireless network data in addition to software to aid the analysis of such data for the wireless network community [75]. However, no such toolkit is available to the NILM community.

3.1.1 Key Contributions

Against this background, we proposed NILMTK[1]; an open source toolkit designed specifically to enable easy access to and comparative analysis of energy disaggregation algorithms across diverse data sets. NILMTK provides a complete pipeline from data sets to accuracy metrics, thereby lowering the entry barrier for researchers to implement a new algorithm and compare its performance against the current state of the art. NILMTK has been:

- Released as open source software (with documentation[2]) 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.
- Designed using a modular structure, therefore allowing researchers to reuse or replace individual components as required. The API design is influenced by `scikit-learn` [88], which is a machine learning library in Python, well known for its consistent API and complete documentation.
- 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.

The contributions of NILMTK are summarised as follows:

[1]Code: `http://github.com/nilmtk/nilmtk`
[2]Documentation: `http://nilmtk.github.io/nilmtk`

- We propose NILMTK-DF (data format), the standard energy disaggregation data structure used by our toolkit. NILMTK-DF is modelled loosely on the REDD data set format [74] to allow easy adoption within the community. Furthermore, we provide parsers from six existing data sets into our proposed NILMTK-DF format.
- We provide statistical and diagnostic functions which provide a detailed understanding of each data set. We also provide 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 [50], and second an approach based on the factorial hidden Markov model [74, 68]. 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 enables the evaluation of any disaggregation algorithm compatible with NILMTK. This allows the performance of a disaggregation algorithm to be evaluated for a range of use cases.

It must be mentioned that NILMTK has been extensively tested only on datasets having sampling rates of 1 Hz or less. While fundamentally, NILMTK can handle time-series data at any resolution, it is not fine-tuned to high frequency data. We know of some ongoing work (by other researchers) that involves using BLUED data (high-frequency) in NILMTK but the findings have not yet been published.

3.2 General Purpose Toolkits

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 [88] and `GraphLab` is a machine learning and data mining toolkit written in C++ [77]. While such toolkits provide generic implementations of 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.

3.3 Energy Disaggregation 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 $x_t^{(n)} \in \mathrm{Z}_{>0}$ to represent the ground truth state, and $\hat{x}_t^{(n)}$ to represent the appliance state estimated by a disaggregation algorithm.

3.4 NILMTK

We designed NILMTK with two core use cases in mind. First, it should enable the analysis of existing data sets and algorithms. Second, it should provide a simple interface for the addition of new data sets and algorithms. To do so, 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 (e.g. `Django`). Furthermore, Python allows easy deployment in diverse environments including academic settings and is increasingly being used for data science.

Figure 3-1 presents the NILMTK pipeline from the import of data sets to the evaluation of various disaggregation algorithms over various metrics. In the remainder of this section we discuss each module of the pipeline: the NILMTK data format, the data set diagnostics and statistics, preprocessing, disaggregation, model import and export and finally we describe accuracy metrics.

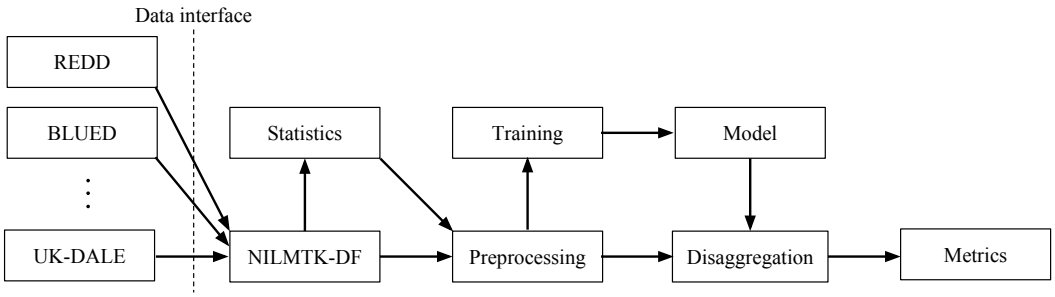

Figure 3-1: NILMTK pipeline. At each stage of the pipeline, results and data can be stored to or loaded from disk.

3.4.1 NILMTK-DF Data Format

Motivated by our discussion in Section 1.3.3 of the wide differences between multiple data sets released in the public domain, we propose NILMTK-DF; a common data set format inspired by the REDD format [74], into which existing data sets can be converted. NILMTK currently includes importers for the following six data sets: REDD, Smart*, Pecan Street, iAWE, AMPds and UK-DALE. BLUED was excluded due to the lack of sub-metered power data, the Tracebase data set was excluded due to the lack of household aggregate power data and HES was excluded due to time constraints.

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 that such additional sensor and metadata information may help enhance NILM prediction [93].

Another important feature of our format is the standardisation of nomenclature. 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 [63].

In addition, NILMTK allows rich metadata to be associated with a household, 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 house (to enable weather data to be retrieved), 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 and whether a specific lamp is dimmable. Our full NILM Metadata schema is described in [63].

Through such a combination of metadata and standard nomenclature, NILMTK allows for analysis of appliance data across multiple data sets. For example, users can perform queries such as: 'what is the energy consumption of refrigerators in the USA compared to the UK?'.

We have defined a common interface for data set importers which, combined with the definition of our in-memory data structures, enables developers to easily add new data set importers to NILMTK.

3.4.2 Data Set Statistics

Distinct from *diagnostic* statistics, NILMTK also provides functions for exploring appliance usage, e.g.:

Proportion of energy sub-metered: Data sets rarely sub-meter every appliance or circuit, and as a result it is useful to quantify the proportion of total energy measured by sub-metered channels. Prior to calculating this statistic, all gaps present in the mains recordings are masked out of each sub-metered channel, and therefore any additional missing sub-meter data is assumed to be due to the meter and load being switched off.

Further functions are listed in in the statistics section of the online documentation.[3]

[3] `http://nilmtk.github.io/nilmtk/stats.html`

3.4.3 Preprocessing of Data Sets

To mitigate the problems with different data sets, some of which were presented in Section **??**, NILMTK provides several preprocessing functions, including:

Downsample: As seen in Table 1.1, the sampling rate of appliance monitors varies from 0.008 Hz to 16 kHz across the data sets. The downsample preprocessor down-samples data sets to a specified frequency using aggregation functions such as mean, mode and median.

Voltage normalisation: The data sets presented in Table 1.1 have been collected from different countries, where voltage fluctuations vary widely. Batra et al. showed voltage fluctuates from 180-250 V in the iAWE data set collected in India [15], while the voltage in the Smart* data set varies across the range 118-123 V. Hart suggested to account for these voltage fluctuations as they can significantly impact power draw [50]. Therefore, NILMTK provides a voltage normalisation function based on Hart's equation:

$$Power_{normalised} = \left(\frac{Voltage_{nominal}}{Voltage_{observed}} \right)^2 \times Power_{observed} \tag{3.1}$$

Top-k appliances: It is often advantageous to model the top-k energy consuming appliances instead of all appliances for the following three reasons. First, the disaggregation of such appliances provides the most value. Second, such appliances contribute the most salient features, and therefore the remaining appliances can be considered to contribute only noise. Third, each additional modelled appliance might contribute significantly to the complexity of the disaggregation task. Therefore, NILMTK provides a function to identify the top-k energy consuming appliances.

NILMTK also provides preprocessing functions for fixing other common issues with these data sets, such as: (i) interpolating small periods of missing data when appliance sensors did not report readings, (ii) filtering out implausible values (such as readings where observed voltage is more than twice the rated voltage) and (iii) filtering out appliance data when mains data is missing.

Each data set importer defines a `preprocess` function which runs the necessary

preprocessing functions to clean the specific data set.

A detailed account of preprocessing functions supported by NILMTK can be found in the online documentation.[4]

3.4.4 Training and Disaggregation Algorithms

NILMTK provides implementations of two common benchmark disaggregation algorithms: combinatorial optimisation (CO) and factorial hidden Markov model (FHMM). CO was proposed by Hart in his seminal work [50], while techniques based on extensions of the FHMM have been proposed more recently [74, 68]. 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 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, subject to a set of appliance models.

$$\hat{x}_t^{(n)} = argmin_{\hat{x}_t^{(n)}} \left| \bar{y}_t - \sum_{n=1}^{N} \hat{y}_t^{(n)} \right| \tag{3.2}$$

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 appliance states. Since the complexity of CO is exponential in the number of appliances, the approach is only computationally tractable for a small number of modelled appliances.

Factorial Hidden Markov Model: The power demand of each appliance can be modelled as the observed value of a 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

[4] `http://nilmtk.github.io/nilmtk/preprocessing.html`

HMM [41] 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. Such a FHMM model has three parameters: (i) prior probability (π) containing K^N entries, (ii) transition matrix (A) containing $K^N \times K^N$ or K^{2N} entries, and (iii) emission matrix (B) containing $2K^N$ entries. The complexity of exact disaggregation for such a model is $O(TK^{2N})$, and as a result FHMMs scale even worse than CO. From an implementation perspective, even storing (or computing) A for 14 appliances with two states each consumes 8 GB of RAM. Hence, we propose to validate FHMMs on preprocessed data where the top-k appliances are modelled, and appliances contributing less than a given threshold are discarded. However, it should be noted that more efficient pseudo-time algorithms could alternatively be used for inference over both CO and FHMM.

For algorithms such as FHMMs, it is necessary to model the relationships amongst consecutive samples. Thus, NILMTK provides facilities for dividing data into continuous sets for training and testing. While we have discussed supervised and non-event based algorithms here, NILMTK also supports event based and unsupervised approaches.

3.4.5 Appliance Model Import and Export

Many approaches require sub-metered power data to be collected for training purposes from the same household in which disaggregation is to be performed. However, such data is costly and intrusive to collect, and therefore is unlikely to be available in a large-scale deployment of a NILM system. As a result, recent research has proposed training methods which do not require sub-metered power data to be collected from each household [68, 86]. To provide a clear interface between training and disaggregation algorithms, NILMTK provides a *model* module which encapsulates the results of the training module required by the disaggregation module. Each implementation of the module must provide import and export functions to interface with a JSON file for persistent model storage. NILMTK currently includes importers and exporters for both the FHMM and CO approaches described in Section 3.4.4.

Data set	Number of appliances	Percentage energy sub-metered	Dropout rate (percent) ignoring gaps	Mains up-time per house (days)	Percentage up-time
REDD	16	71	10	18	40
Smart*	25	86	0	88	96
AMPds	20	97	0	364	100
iAWE	10	48	8	47	93
UK-DALE	12	48	7	102	84

Table 3.1: Summary (median) of data set results calculated by the diagnostic and statistical functions in NILMTK. Each cell represents the range of values across all households per data set.

3.4.6 Accuracy Metrics

A range of accuracy metrics are required due to the diversity of application areas of energy disaggregation research. To satisfy this requirement, NILMTK provides a set of metrics which combines both general detection metrics and those specific to energy disaggregation. We now give a brief description of each metric implemented in NILMTK along with its mathematical definition.

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

$$\left| \sum_t y_t^{(n)} - \sum_t \hat{y}_t^{(n)} \right| \tag{3.3}$$

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

$$\sum_n \min \left(\frac{\sum_n y_t^{(n)}}{\sum_{n,t} y_t^{(n)}}, \frac{\sum_n \hat{y}_t^{(n)}}{\sum_{n,t} \hat{y}_t^{(n)}} \right) \tag{3.4}$$

Normalised error in assigned power: The sum of the differences between the assigned power and actual power of appliance n in each time slice t, normalised by the appliance's total energy consumption.

$$\frac{\sum_t \left| y_t^{(n)} - \hat{y}_t^{(n)} \right|}{\sum_t y_t^{(n)}} \tag{3.5}$$

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

$$\sqrt{\frac{1}{T}\sum_{t}\left(y_t^{(n)} - \hat{y}_t^{(n)}\right)^2} \tag{3.6}$$

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

True positives, False positives, False negatives, True negatives: 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).

$$TP^{(n)} = \sum_{t} \text{AND}\left(x_t^{(n)} = on, \hat{x}_t^{(n)} = on\right) \tag{3.7}$$

$$FP^{(n)} = \sum_{t} \text{AND}\left(x_t^{(n)} = off, \hat{x}_t^{(n)} = on\right) \tag{3.8}$$

$$FN^{(n)} = \sum_{t} \text{AND}\left(x_t^{(n)} = on, \hat{x}_t^{(n)} = off\right) \tag{3.9}$$

$$TN^{(n)} = \sum_{t} \text{AND}\left(x_t^{(n)} = off, \hat{x}_t^{(n)} = off\right) \tag{3.10}$$

True/False positive rate: The fraction of time slices in which an appliance was correctly predicted to be on that it was actually on (TPR), and the fraction of time slices in which the appliance was incorrectly predicted to be on that it was actually off (FPR). We omit appliance indices n in the following metrics for clarity.

$$TPR = \frac{TP}{(TP + FN)} \tag{3.11}$$

$$FPR = \frac{FP}{(FP + TN)} \tag{3.12}$$

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

which the appliance was correctly predicted to be on that it was actually on (Recall).

$$Precision = \frac{TP}{(TP + FP)} \tag{3.13}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{3.14}$$

F-score: The harmonic mean of precision and recall.

$$F\text{-}score = \frac{2.Precision.Recall}{Precision + Recall} \tag{3.15}$$

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

$$HammingLoss = \frac{1}{T}\sum_{t}\frac{1}{N}\sum_{n}\text{XOR}\left(x_t^{(n)}, \hat{x}_t^{(n)}\right) \tag{3.16}$$

3.5 Example Data Flow

Having described the features of the NILMTK pipeline, we will now look into an example to illustrate the flow of data in the same. We assume that a new data set called `SampleDS` has been made available. This data set contains 1 Hz appliance and aggregate data from 5 homes in CSV format. The data set importer is a set of scripts that convert the raw data into NILMTK-DF. It will ensure that the appliances used have labels consistent with the NILMTK terminology. The statistics stage will be used to calculate various statistics such as the percentage of energy submetered. Homes having small amount of energy submetered should probably be discarded from the analysis. Also, homes having a high amount of data loss should be discarded. In the preprocessing step, we can resample the data. For instance, in accordance with smart metering standards, we may choose to use the data at minutely resolution instead of the 1 Hz resolution. This is handled by the preprocessing stage. In the training stage, we use existing benchmark algorithms to train on the top-5 appliance by energy consumption. We export the trained model to JSON so that we can use

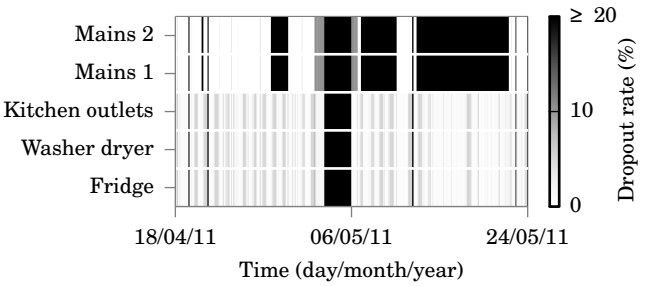

Figure 3-2: Lost samples per hour from a representative subset of channels in REDD house 1.

it in a web application. Finally, we use the trained model to disaggregate on the mains data from the data set. The procedure was done by using train-test split as required by the experiments. Finally, a bunch of metrics as per the application were computed on the disaggregated data. Some applications only care about the state of the appliance. For such applications, one may use metrics such as F-score. For some applications, the error in prediction may be important, and for them we can use metrics like RMS error.

3.6 Evaluation

We now demonstrate several examples of the rich analyses supported by NILMTK. First, we diagnose some common (and inevitable) issues in a selection of data sets. Second, we show various patterns of appliance usage. Third, 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. Fourth, we present summary performance results of the two benchmark algorithms included in NILMTK across six data sets using a number of accuracy metrics. Finally, we present detailed results of these algorithms for a single data set, and discuss their performance for different appliances.

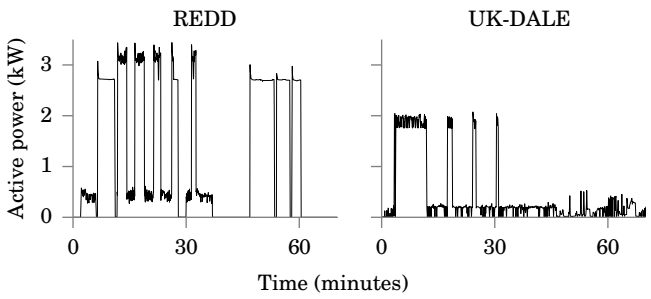

Figure 3-3: Comparison of power draw of washing machines in one house from REDD (USA) and UK-DALE.

3.6.1 Data Set Diagnostics

Table 3.1 shows a selection of diagnostic and statistical functions (defined in Section **??** and 3.4.2) computed by NILMTK across six public data sets. BLUED, Tracebase and HES were not included for the same reasons as in Section 3.4.1. The table illustrates that AMPds used a robust recording platform because it has a percentage up-time of 100%, a dropout rate of zero and 97% of the energy recorded by the mains channel was captured by the sub-meters. Similarly, Pecan Street has an up-time of 100% and zero dropout rate. However, two homes in the Pecan Street data registered a proportion of energy sub-metered of over 100%. This indicates that some overlap exists between the metered channels, and as a result some appliances are metered by multiple channels. This illustrates the importance of data set metadata (proposed as part of NILMTK-DF in Section 3.4.1) describing the basic mains wiring.

Figure 3-2 shows the distribution of missing samples for REDD house 1. From this we can see that each mains recording channel has four large gaps (the solid black blocks) where the sensors are off. The sub-metered channels have only one large gap. Ignoring this gap and focusing on the time periods where the sensors are recording, we see numerous periods where the dropout rate is around 10%. Such issues are by no means unique to REDD and are crucial to diagnose before data sets can be used for the evaluation of disaggregation algorithms or for data set statistics.

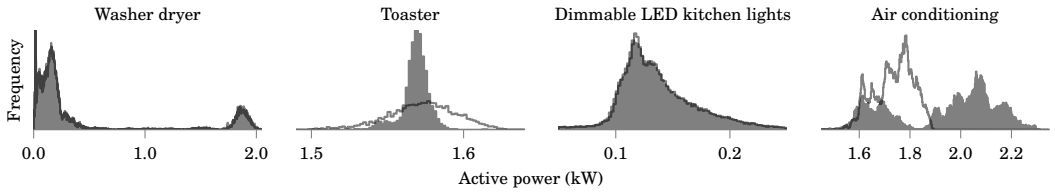

Figure 3-4: Histograms of power consumption. The filled grey plots show histograms of normalised power. The thin, grey, semi-transparent lines drawn over the filled plots show histograms of un-normalised power.

3.6.2 Data Set Statistics

Energy disaggregation systems must model individual appliances. Hence, as well as diagnosing technical issues with each data set, NILMTK also provides functions to visualise patterns of behaviour recorded in each data set. For example, different appliances draw a different amount of power (e.g. a toaster draws approximately 1.57 kW), are used at different times of day (e.g. the TV is usually on in the evening) and have different correlations with external factors such as weather (e.g. lower outside temperature implies more usage of electric heating). Furthermore, load profiles of different appliances of the same type can vary considerably, especially appliances from different countries (e.g. the two washing machine profiles in Figure 3-3). Some disaggregation systems benefit by capturing these patterns (for example, the conditional factorial hidden Markov model (CFHMM) [68] can model the influence of time of day on appliance usage). In the following sections, we present examples of how such information can be extracted from existing data sets using NILMTK, covering the distribution of appliance power demands (Section 3.6.3), usage patterns (Section 3.6.4) and external dependencies (Section 3.6.5).

3.6.3 Appliance power demands

Figure 3-4 displays histograms of the distribution of powers used by a selection of appliances (the washer dryer, toaster and dimmable LED kitchen lights are from UK-DALE house 1; the air conditioning unit is from iAWE). Appliances such as toasters and kettles tend to have just two possible power states: on and off. This simplicity makes them amenable to be modelled by, for example, Markov chains with only two

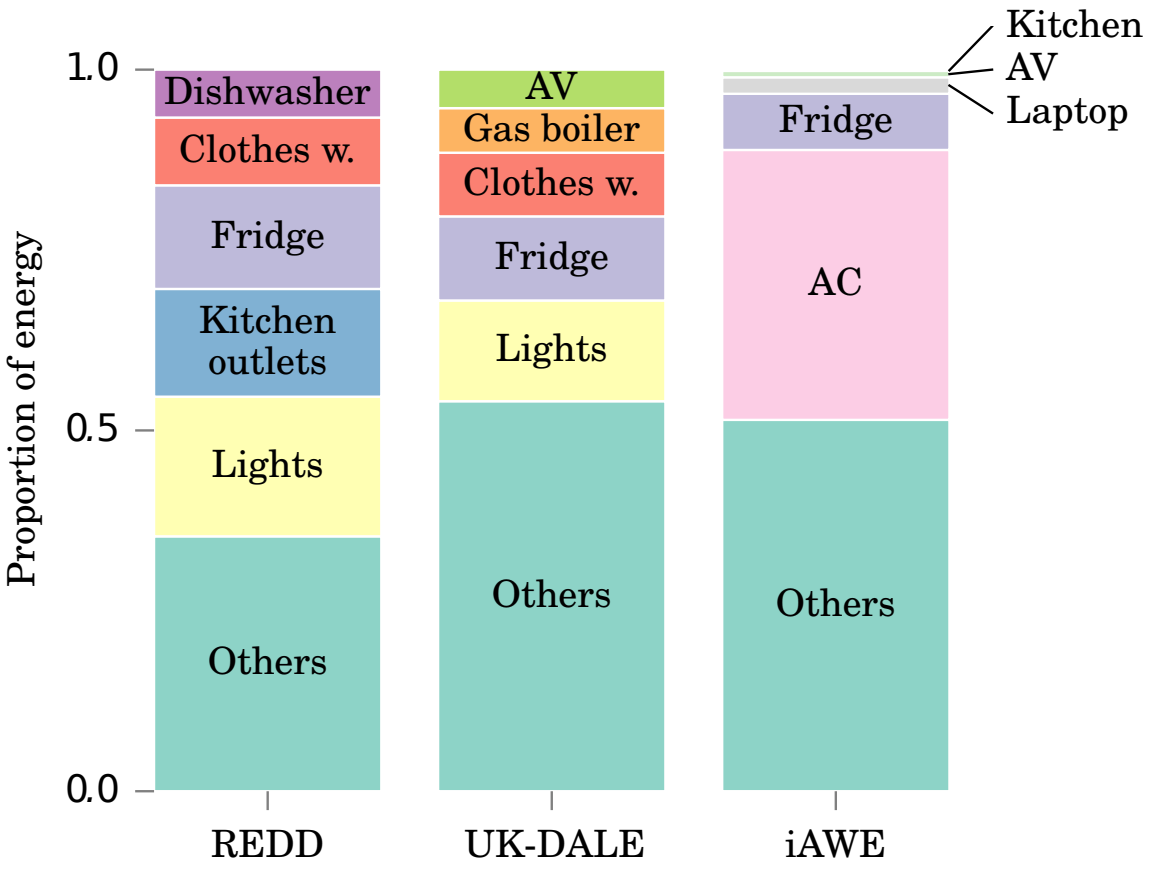

Figure 3-5: Top five appliances in terms of the proportion of the total energy used in a single house (house 1) in each of REDD (USA), iAWE (India) and UK-DALE.

states per chain. In contrast, more complex appliances such as washing machines, vacuum cleaners and computers often have many more states.

Figure 3-5 shows examples of how the proportion of energy use per appliance varies between countries. It can seen that the REDD and UK-DALE households share some similarities in the breakdown of household energy consumption. In contrast, the iAWE house shows a vastly different energy breakdown. For example, the house recorded in India for the iAWE data set has two air conditioning units which account for almost half of the household's energy consumption, whilst the example household from the UK-DALE data set does not even contain an air conditioner.

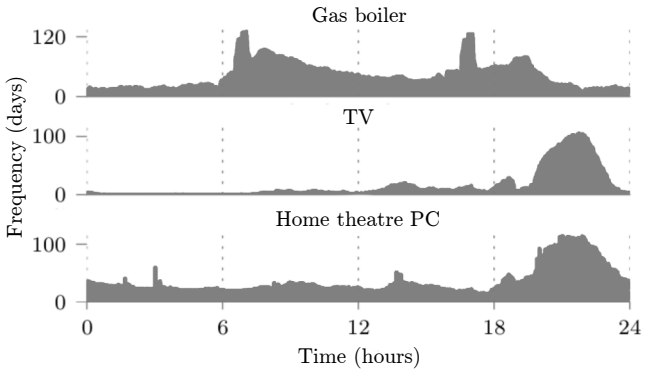

Figure 3-6: Daily appliance usage histograms of three appliances over 120 days from UK-DALE house 1.

3.6.4 Appliance usage patterns

Figure 3-6 shows histograms which represent usage patterns for three appliances over an average day, from which strong similarities between groups of appliances can be seen. For example, the usage patterns of the TV and Home theatre PC are very similar because the Home theatre PC is the only video source for the TV. In contrast, the boiler has a usage pattern which occurs as a result of the household's occupancy pattern and hot water timer in mornings and evenings.

3.6.5 Appliance correlations with weather

Previous studies have shown correlations between temperature and heating/cooling demand in Australia [91] and between temperature and total household demand in the USA [60]. Such correlations could be used by a NILM system to refine its appliance usage estimates [101].

Figure 3-7 shows correlations between boiler usage and maximum temperature (appliance data from UK-DALE house 1, temperature data from UK Met Office). 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.

Data set	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*	3.40	46.34	0.39	1.85	3.10	2.71	0.50	0.66	0.53	0.61
Pecan Street	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
UK-DALE	1.06	11.42	0.10	0.52	3.66	3.67	0.81	0.80	0.38	0.38

Table 3.2: Comparison of CO and FHMM across multiple data sets.

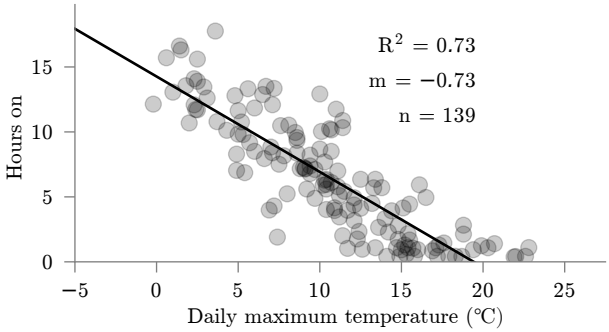

Figure 3-7: Linear regression showing correlation between gas boiler usage and external temperature. R^2 denotes the coefficient of determination, m is the gradient of the regression line and n is the number of data-points (days) used in the regression.

3.6.6 Voltage Normalisation

Normalisation can be used to minimise the effect of voltage fluctuations in a household's aggregate power. Figure 3-4 shows histograms for both the normalised and un-normalised appliance power consumption. Normalisation produces a noticeably tighter power distribution for linear resistive appliances such as the toaster, although it has little effect on constant power appliances, such as the washer dryer or LED kitchen ceiling lights. Moreover, for non-linear appliances such as the air conditioner, normalisation increases the variance in power draw. This is in conformance with work by Hart [50] which proposed a modified approach to normalisation:

$$Power_{normalised} = \left(\frac{Voltage_{nominal}}{Voltage_{observed}} \right)^{\beta} \times Power_{observed} \tag{3.17}$$

For linear appliances such as the toaster, $\beta = 2$, whereas for appliances such as fridge, Hart found $\beta = 0.7$. Thus, we believe the benefit of voltage normalisation is dependent on the proportion of resistive loads in a household.

3.6.7 Disaggregation Across Data Sets

We now compare the disaggregation results across the first house of six publicly available data sets. Again, BLUED, Tracebase and HES were not included for the same reasons as in Section 3.4.1. Since all the data sets were collected over different durations, we used the first half of the samples for training and the remaining half for disaggregation across all data sets. Further, we preprocessed the REDD, UK-DALE, Smart* and iAWE data sets to 1 minute frequency using the down-sampling filter (Section 3.4.3) to account for different aggregate and mains data sampling frequencies and compensating for intermittent lost data packets. The small gaps in REDD, UK-DALE, SMART* and iAWE were interpolated, while the time periods where either the mains data or appliance data were missing were ignored. AMPds and the Pecan Street data did not require any preprocessing.

Since both CO and FHMM have exponential computational complexity in the number of appliances, we model only those appliances whose total energy contribution was greater than 5%. Across all the data sets, the appliances which contribute more than 5% of the aggregate include HVAC appliances such as the air conditioner and electric heating, and appliances which are used throughout the day such as the fridge. We model all appliances using two states (on and off) across our analyses, although it should be noted that any number of states could be used. However, our experiments are intended to demonstrate a fair comparison of the benchmark algorithms, rather than a fully optimised version of either approach. We compare the disaggregation performance of CO and FHMM across the following three metrics defined in Section 3.4.6: (i) fraction of total energy assigned correctly (FTE), (ii) normalised error in assigned power (NEP) and (iii) F-score. These metrics were chosen because they have been used most often in prior NILM work. F-score and FTE vary between 0 and 1, while NEP can take any non-negative value. Preferable performance is indicated by a low NEP and a high FTE and F-score. The evaluation was performed on a laptop with a 2.3 GHz i7 processor and 8 GB RAM running Linux. We fixed the random seed for experiment repeatability, the details of which

can be found on the project github page.

Table 3.2 summarises the results of the two algorithms across the six data sets. It can be observed that FHMM performance is superior to CO performance across the three metrics for REDD, Smart* and AMPds. This confirms the theoretical foundations proposed by Hart [50]; that CO is highly sensitive to small variations in the aggregate load. The FHMM approach overcomes these shortcomings 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 Street and UK-DALE across all metrics. This is likely due to the fact that very few appliances contribute more than 5% of the household aggregate load in the selected households in these data sets. For instance, space heating contributes very significantly (about 60% for a single air conditioner which has a power draw of 2.7 kW in the Pecan Street house and about 35% across two air conditioners having a power draw of 1.8 kW and 1.6 kW respectively in iAWE). 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. In the UK-DALE house the washing machine was one of the appliances contributing more than 5% of the household aggregate load, which brought down overall metrics across both approaches.

Another important aspect to consider is the time required for training and disaggregation, again reported in Table 3.2. These timings confirm the fact that CO is exponentially quicker than FHMM. This raises an interesting insight: in households such as the ones used from Pecan Street and iAWE in the above analysis, it may be beneficial to use CO over a FHMM owing to the reduced amount of time required for training and disaggregation, even though FHMMs are in general considered to be more powerful. It should be noted that the greater amount of time required to train and disaggregate the AMPds data is a result of the data set containing one year of data, as opposed to the Pecan Street data set which contains one week of data, as shown by Table 1.1.

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 3.3: Comparison of CO and FHMM across different appliances in iAWE data set.

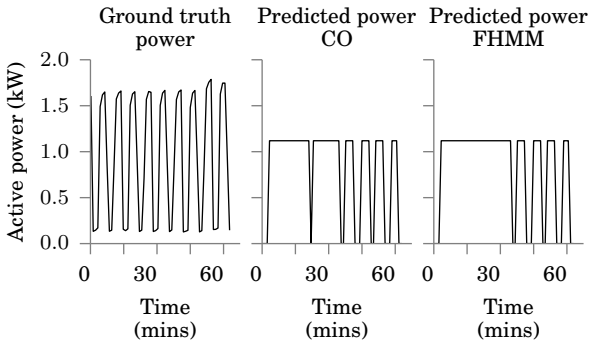

Figure 3-8: Predicted power (CO and FHMM) with ground truth for air conditioner 2 in the iAWE data set.

3.6.8 Detailed Disaggregation Results

Having compared disaggregation results across different data sets, we now give a detailed discussion of disaggregation results across different appliances for a single house in the iAWE data set. The iAWE data set was chosen for this experiment as the authors provided metadata such as set temperature of air conditioners and other occupant patterns. Table 3.3 shows the disaggregation performance across the top six energy consuming appliances, in which each appliance is modelled using two states as before. It can be seen that CO and FHMM report similar performance across all appliances. We observe that the results for appliances such as the washing machine and switch mode power supply based appliances such as laptop and entertainment unit (television) are much worse when compared to HVAC loads like air conditioners across both metrics. Furthermore, prior literature shows that complex appliances

such as washing machines are hard to model [7].

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 an external temperature of roughly $30 - 35$ °C, 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 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 3-8 shows how both FHMM and CO are able to detect on and off events of air conditioner 2. Since air conditioner 2 spent a considerable amount of time in the intermediate state, the learnt two state model is less appropriate in comparison to the two state model used for air conditioner 1. This can be further seen in the figure, where we observe that both FHMM and CO learn a much lower power level of around 1.1 kW, in comparison to the rated power of around 1.6 kW. We believe that this could be corrected by learning a three state model for this air conditioner, which comes at a cost of increased training and disaggregation computational and memory requirements.

3.7 NILMTK for large data sets

NILMTK was originally designed to handle the relatively small data sets (less than 10 households) which were available at the time of release. As such, the toolkit was not suitable for use with larger data sets (hundreds of households) which have been released since (e.g. Dataport data set). As a result, it was not possible to evaluate energy disaggregation approaches at a sufficient scale so as to investigate the extent of their generality. To address this shortcoming, we presented a new release of the toolkit (NILMTK v0.2) [62] which is able to evaluate energy disaggregation algorithms using arbitrarily large data sets. Rather than loading the entire data set into memory, the aggregate data is loaded in chunks and the output of the disaggregation algorithm is saved to disk chunk-by-chunk (as shown in Figure 3-9. As a result, we are able to

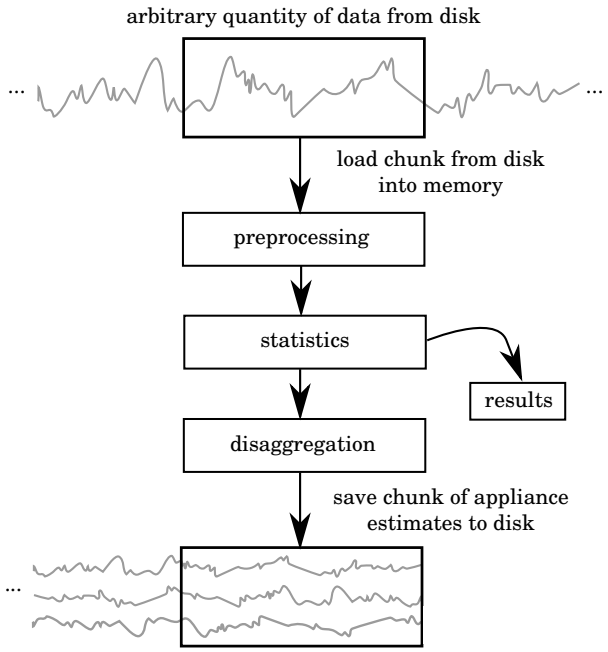

Figure 3-9: NILMTK v0.2 can process an arbitrary quantity of data by loading data from disk in chunks. This figure illustrates the loading of a chunk of aggregate data from disk (top) and then pushing this chunk through a processing pipeline which ends in saving appliance estimates to disk chunk-by-chunk.

demonstrate data set statistics and disaggregation for the Dataport data set, which contained 239 households of aggregate and individual appliance power data at the time of NILMTK current version. In addition to scalability improvements, the current version also includes support for a rich data set metadata description format, as well as a number of usability improvements and many software design improvements.

3.8 Summary

Despite three decades of research, it was virtually impossible to compare energy disaggregation literature. This was due to three key problems: 1) different data sets used, 2) lack of reference benchmark algorithms, and 3) variety of accuracy metrics used. We presented 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. This work was the first research to compare multiple disaggregation approaches across multiple publicly available data sets. Our

toolkit includes parsers for a range of existing data sets, a collection of preprocessing algorithms, a set of statistics for describing data sets, two reference benchmark disaggregation algorithms and a suite of accuracy metrics. NILMTK has been well received by the community as evidenced by multiple data sets and algorithms contributed by the community, and awards in international conferences.