Energy Disambiguation

Submitted as Final Thesis - SIT724

SUBMISSION DATE
T2-2022

Mohamed Arshad Fazeer
STUDENT ID 218692903
COURSE - Bachelor of Software Engineering Honours (S464)

Supervised by: Jean-Guy Schneider/Niroshinie Fernando

Abstract

In recent years there has been an increased usage of affordable individual smart plugs as opposed to purchasing a smart metre for the entire house, this is because you would have access to the energy consumption of each individual device. It is common for almost all households to use extension cords or extension adapters to connect multiple appliances to one electrical socket. This is where the problem arises with smart plugs and what this report is based on, the purpose of individual device consumption is lost when multiple devices are connected to a single smart plug, as it would return the aggregated energy consumption of all the devices connected through the smart plug. To address this challenge, deep research and analysis has been conducted to find the optimum solution for disaggregating power from an aggregated source. There are 2 approaches displayed in this paper, Graph Signal Processing that uses pattern matching with a signature database to isolate devices from the combined power source, where as Pruned NILM which is a machine learning algorithm that optimises the common NILM approach that is widely used in solving disaggregation of power. For short-term aggregated data, GSP method is the most efficient way since there is no training required and is of low complexity, where as long-term and sophistacated data, it would be essential to perform machine-learning algorithms with Pruned NILM to get accurate results.

Contents

List of Figures

List of Tables

1 Introduction

Energy disambiguation is the process of estimating the power consumption of individual appliances when the aggregate power is collected from a single smart plug connected to a power outlet.

In recent years, the development and conversion of regular homes into ”smart homes” has increased dramatically. This can be ascribed to technologies like the Internet of Things and smart plugs in specific. This is due to the competency of smart plugs being able to give enough energy consumption information for the development and deployment of systems to manage energy consumption, they are at the heart of transforming an ordinary home into a relatively smart home. [23]. Common households use extension cords or extension adapters to connect multiple devices using a single power source. The purpose of individual device consumption is lost when multiple devices are connected to a single smart plug, as it would return the aggregated energy consumption of all the devices that is connected through the smart plug.

1.1 What is Energy Disambiguation?

Energy disambiguation more commonly referred to as energy disaggregation has been in the engineering research community for a long time for the purpose of decreasing energy consumption. This is a developing concern since energy resources are limited, and the global energy consumption is expected to double by 2030 with significant environmental consequences[36] . Most researchers have applied some sort of machine learning technique to solve the problem of diaggregation, some of the earliest being in the 1980s-90s such as [30],[11] that would be able to capture when a device is turned on or off using the 'edges' or nodes, subsequent studies focused more on determining the device signatures and footprints to find the actual power consumption and disaggregating aggregated data through machine learning - [7],[28],[25], [20].

1.2 Objective

There are various approaches that can be taken to solve the problem of disaggregation, commonly done by machine learning algorithms for the purpose of assisting households

minimise their energy consumption. However the 2 approaches presented are distinct in which one uses pattern matching and does not require any machine learning rather it relies on Graph Signal Processing(GSP). The second method is a variant of the commonly used NILM approach known as PrunedNILM. The method follows a traditional machine learning approach using sequence-to-point learning with neural networks.

A lot of prior research has focused on smart homes and smart meters that record data of an entire household or building where as the first method which is GSP focuses more on desegregating aggregated power data collected from random devices through the means of a single smart plug, the process goes through a data collection application - that collects data of power consumption through the smart plug, a smoothing process - to clean the data and achieve better trend lines, and then through pattern recognition techniques to be able to isolate the power used from each device off a grouped data-set.

Previous work has demonstrated that one of the most effective methods for combating NILM is seq2point learning [35]. This method employs a series of total power statistics to trace the power of target single appliance usages at the midpoint of the power data frame. However, the models created using this technique can include over 30 million weights which calls for substantial amounts of resources and time in order to get results for disaggregation. Therefore a solution to decrease the amount of weights is by pruning the weights and creating a reduced architecture which results in much lower times and resources while also keeping the accuracy of the results in check.

1.3 Structure

Section 2 consists of detailed review of the existing literature on the topic of Energy Disambiguation. Section 3 presents the research design which is the project goals, experimental phases and data requirements. Section 4 describes the approach and the technical details of artefact development which is based on the power consumption data collected and the smoothing algorithms used in phase 1. GSP and Pruned-NILM disaggregation methods are discussed and developed in phase 2 Section 5 evaluates the artefacts and analyses the smoothing out of data, the GSP results, the Pruned-NILM results and a detailed performance matrix for accuracy. Section 6 concludes the report and gives an overview of the future work

2 Literature Review

There are numerous papers on disaggregation of energy from a single house or a set of houses, This literature review consists of 3 parts: Smoothing Techniques 4.2.1, Related work on Energy Disaggregation 2.2, and Methods of Disaggregation 2.3 - focuses more on the method itself rather than the overview of their project and finally with the literature gaps and possible furture research direction. 2.4

2.1 Smoothing Techniques

Kaleg and Hapid, processed observational data using a data smoothing approach in order to see trends and patterns. The data was based on observations from real-world electric vehicle driving tests, namely data on the electrical power supplied and received by the vehicle's battery pack. During the real-life driving test, a data logger collected the electrical power data. Furthermore, the data smoothing was evaluated in order to determine the repercussions of its application. The Simple Moving Average (SMA), Single Exponential Smoothing (SES), and Double Exponential Smoothing (DES) were three data smoothing approaches that were investigated as a pilot study.Furthermore, the Mean Arctangent Absolute Percentage Error was used to assess the data smoothing results (MAAPE). They then concluded that for the purpose of knowing trends of the particular study, SMA with a period constant of 250 shows the best results. [15]

Chowdhury and Tahmid, utilized exponential smoothing to reduce the noise in radar signal waves, Exponential smoothing is a technique for smoothing time series data using an exponential window function. [8]

2.2 Related Work on Disaggregation

Pandey and Karypis, presents an extension of Powerlet based energy disaggregation (PED). PED is a dictionary learning method that predicts different power consumption patterns by capturing the combination of each appliance as representatives (used as the dictionary atoms). It does however have difficulty in distinguishing which appliances

are in use when the devices have similar power consumption. For handling these cases, they take into consideration that rather than simply co-occurring together, it could be that the devices are operating in different modes to accomplish a certain task. They developed algorithms that can extract the concurrent operation of distinct devices without utilising any further information to distinguish between two identical power consuming devices. Their methods find the representatives among the device set's aggregated power consumption. If devices in the specific set co-occur in a certain operation mode then that operation mode's representative power consumption becomes one of the retrieved representatives.The device set is recursively deconstructed level by level into 2 equal partitions (or one partition containing one more device in the event that the total number of devices in odd). The devices are organised in this fashion to form a binary tree, which results in a hierarchical clustering of the devices. Each individual device is represented by the leaves of the tree, whereas subsets of devices are represented by interior nodes whose hierarchy rises as it progresses up the tree. Instead of disaggregating all of the devices at once, the tasks were deconstructed as a recursive one, in which the power consumption of subsets of devices was estimated at each level from top to bottom. They investigated two different heuristics for composing the hierarchical tree, Greedy based Device Decomposition Method (GDDM) and Dynamic Programming based Device Decomposition Method (DPDDM), to which GDDM had performed significantly better. [25].

Sinha, Spoorthy, Kharuna and Chandra, attempted to build a generic framework for energy disaggregation by combining the modelling, measurement, and calibration procedures, as well as the inference on sparse data models. They presented a study of load disaggregation of home appliances using sparse data models(data dependant dictionaries) learned from individual appliances based on aggregate power measurements. Data generated by physical processes or systems had been demonstrated to be valuable for representation, inference, storage, and communication using sparse data models. These data models were used to create measurement models that were in turn calibrated to the data. On the basis of sparse data models, inferences on the data include a) event detection and b) feature estimation. Model parameters were found that can be changed to accomplish a balance between data approximation error within a class and data classification accuracy across classes. The results of this modelling and optimisation approach applied to the REDD data set reveal that it is suitable for inferring individual household appliance power usage from aggregate power measurements. [29]

Koutitas and Tassiulas, presented a non-intrusive load disaggregation algorithm that

considers low frequency data sampling of active power readings and was based on fuzzy logic and pattern recognition. For decision-making, the suggested solution considers an n-dimensional space that includes appliance properties, pulse characteristics, user activity, and external conditions. The user was required to give information on the number, type, and labelled power value of the appliances in the residential unit in the suggested algorithm. The proposed approach was tested in a genuine smart meter environment using Kimatica Ltd's low-cost smart metre technology.According to the simultaneous appliance usage in the residential unit, the observed accuracy of this approach ranges from 70% to 100%. .[21]

Hatham and Greetham, proposed a technique by demonstrating the energy usage of a typical dry cleaner over a six-week period. A disaggregation technique was designed based on data collected on site, allowing specific electrical devices or device combinations to be identified from the aggregate energy consumption. This enables the behaviour of selected equipment to be observed in a hypothetical environment, and as a result, control solutions to improve the energy efficiency of the company's operations to be developed. The approach they used to achieve energy disaggregation is recurrence quantification analysis (RQA).The recurrence plot is used to reveal recurrent patterns in a complex time series that were previously concealed. Then, to quantify the recurring structures depicted by these plots, RQA is used, where time dependent RQA variables were also computed. .[12] Similarly Tomkins, Pujara and Getoor proposed a probabilistic energy disaggregation approach for identifying the most likely set of appliances in use. To distinguish between similar appliances, their framework relies on constraints and context. The framework can adapt to multiple types of information and is able to disaggreate extremely coarse power values as well. [33]

Kui Wu, Lei and Tang, approached the problem through a different lens where they aimed to develop a basic, universal model to disaggregate energy without machine learning techniques or auxiliary equipment, They make use of the easily accessible knowledge of appliance such as the appliances' rated power and power deviation which can be retrieved from the user guide of appliances. The sparsity of the switching events was also used to construct a Sparse Switching Event Recovering (SSER) method. By reducing the total variation (TV) of the (sparse) event matrix, SSER can effectively recover the individual energy consumption values from the aggregated ones. Furthermore a Parallel Local Optimization Algorithm (PLOA) was developed to tackle the problem in active epochs of appliance operations in parallel to speed up the process. They compared the performance of their method to that of state-of-the-

art solutions, such as Least Square Estimation (LSE) and iterative Hidden Markov Model(HMM), using real-world trace data. Their technique had a higher overall detection accuracy and a lower overhead, according to the results. [32]

Since most of the energy disaggregation methods are challenging to perform for ordinary customers, Tang, Chen and Wu proposed to implement an online application called a SmartSaver, a consumer-oriented web service that is not only open and free to consumers, but also user-friendly and simple to use for energy disaggregation. The basis of Smart Saver is a sparse switching event recovery model.Smart Saver is a hybrid system that combines client/server (C/S) and browser/server (B/S) architectures. A Data Storage, a Computing Node, and a Web Server form the system's core, - performing data storage and backup, - load data analysis and energy disaggregation, and - task submission and response, respectively. The three components can be mixed or separated, and they can even be virtualized on the cloud.[31]

Farinaccio and Zmeureanu attempted to use pattern recognition to disaggregate energy from total house consumption. Their research reveals that it has the potential to be used in residential constructions. The results show that utilising a pattern recognition approach and only one sensor positioned on the house's main electric entrance. The whole-house electricity consumption may be disaggregated into its key end-uses. However in order to determine their electric properties, it is a necessity that a one-time sub-metering of the target appliances throughout the training phase, which lasted roughly a week .[10]

2.3 Methods of Disaggregation

Powerlet Based Energy Disaggregation PED created the notion of powerlets. It tackles the problem of disaggregation by first splitting the task into 2 phases: collecting each appliance energy consumption and then conducting the actual disaggregation among the extracted patterns. [25] Most devices consists of multiple operation modes, for example a typical hairdryer has low, medium and high heat or switched off state. PED involves a training phase where assumptions are made on each devices' operation mode by mapping it to a vector, referred to as a 'powerlet'. Thus the aim is to build a dictionary for each device using these powerlets. [9] A drawback with this method is that PED would not be able to catch the devices being concurrently run during the

same timeframe.

BOLT stands for Binary Online Factorization Engine, which aims to solve the computational difficulty of real-time inference while also eliminating the fine-tuning of discriminative features for appliance identification. BOLT is a deep learning-inspired algorithm that transforms the challenge of disaggregation into a binary matrix factorization problem. A neural network is used to learn a nonlinear modification of a single period of a phase-aligned current waveform. This nonlinear mapping is limited in such a way that the network learns sub-component waveforms that best explain the aggregate current when combined together. [22]

NILM - Non Intruisive Load Monitoring enables detection of appliances that are ON or OFF, as well as the characteristics for each appliance connected. NILM has been in the works since the 1980's with the work of George Hart[11] His work outlined trials involving extraction of details and transitions between steady-states. Since then various NILM algorithms have then been developed for various problems mainly in signal processing and for the purpose of disaggregation. Some developments in NILM worth mentioning such as [13] in which they use the KNN(K Nearest Neighbor) method that uses machine learning to classify between different appliances. KNN uses a machine learning algorithm based on memory-based learning.In [6] where NILM is used for evaluation and feedback of fridge energy consumption. Another paper that enlightens NILM is in [17] where deep neural networks is embedded with NILM by training lots of data and allowing neurons to pass information to the next neuron. In [4] NILM has been further developed into a toolkit known as NILMTK which is used to enable the analysis of existing data sets and algorithms and to provide a simple interface for the addition of new data sets and algorithms. There have been various modifications done where neural networks are formed with sequence-to-point learning [35], probabilistic pruning to reduce the weights [34], pruning and then using sequence-to-point learning. [3]

Deep neural networks have been found to be a potential strategy for to reduce it by introducing domain knowledge into the model which in turn reduces the identifiability problem.This can be further solved by following the sequence-to-point learning method where the input is a window of the mains and output is a single point of data from the target appliance. It is methodically demonstrated that the convolutional neural networks have the ability to naturally pick up on the target appliances' signatures, which help lessen the identifiability issue in the model.[35]

Pascal, Iosif and Michael presented a different approach to energy disaggregation by making use of elastic matching techniques. The algorithms compares energy consumption signatures with a database of energy consumption frames, this can be referred to as template matching. Elastic matching algorithms do not have a model training process but processes pattern recognition using template matching. Therefore sizable quantity of data is not required to train a model unlike most other energy disambiguation algorithms that make use of machine learning. There were 5 different elastic matching algorithms that were evaluated which were: Dynamic Time Warping, Global Alignment Kernel, Soft Dynamic Time Warping, Minimum Variance Matching and All common Subsequence. The experimental results showed that the minimum variance matching algorithm outperformed all other evaluated matching algorithms. The best performing minimum variance matching algorithm improved the energy disaggregation accuracy by 2.7% when compared to the baseline dynamic time warping algorithm. [27]

2.4 Gaps in existing literature

Despite the latest interest in NILM, empirically comparing NILM studies and fully understanding the state of the art has remained extremely difficult. Reproducibility is being hampered by three issues, which prohibited empirical comparison of NILM algorithms. For starters, it was difficult to judge the generality of NILM techniques because most research was based on a single data set, and researchers would frequently sub-sample data sets to pick specific appliances, and time periods, making experimental results more difficult to replicate. Second, there was no comparison using the same benchmarks, owing to a lack of open source benchmark implementations. As a result, the majority of study articles compared their findings to variations of their core algorithms or simple baseline methods. Thirdly, a vast number of measures were used dependent on the use case under consideration, making it practically impossible to determine the state-of-the-art. [5]

Other gaps include when the appliances are not constant or are in a special scenario, for example a kettle when it was turned on 5 minutes ago would still retain some of the heat so it would take significantly less power than usual, which has not been discussed or taken in to account. Another scenario is when a laptop is 75% charged and is then plugged in, the pattern of power consumption would be different compared to a normal

charging pattern. Adding to that is when 2 devices that have similar patterns and are switched on at the exact same time it would be difficult to analyse and tell if it was 1 device with a high power consumption or 2 separate low power consuming devices that gives an aggregate power equal to 1 high-power consuming device.

3 Research Design & Methodology

The main objective of this research is to find a suitable algorithm to disaggregate individual device signatures from a grouped dataset. This can be performed by analysing the state-of-the art algorithms that have already been tested and then fill up the research gaps by improving or providing possible solutions. This paper includes an in depth analysis of 2 algorithms. First one being Graph Signal Processing which uses a dataset that was collected through a smartplug for 5 days for evaluation, this does not involve any machine learning where as the second algorithm is PrunedNILM which is a machine learning algorithm and evaluation is done by training a publicly available dataset.

3.1 Project Goals

- To examine and evaluate existing literature and identify literature gaps
- Collect data using a smart plug and smoothen the data to remove any outliers and be able to perform energy disaggregation methods in an efficient manner.
- Construct a design and evaluation plan that aims to address the main research questions that has been established in the report

3.2 Experimental Phases

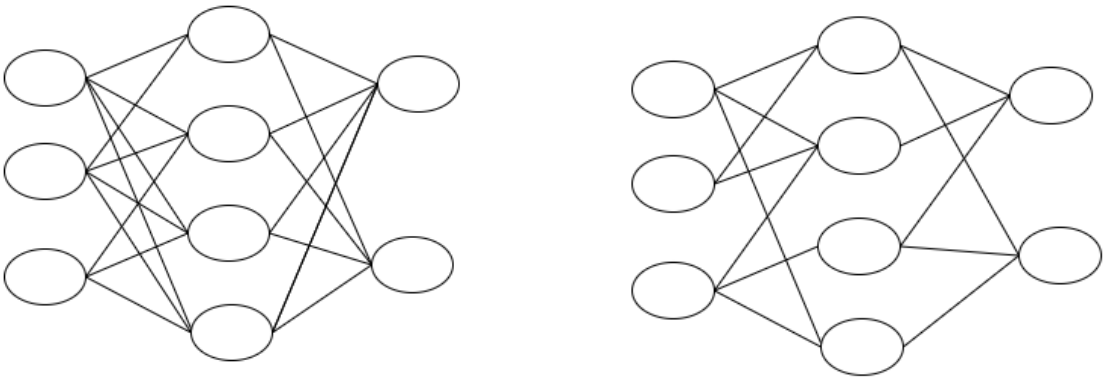
This research is divided into 2 phases, the first phase is where detailed literature review took place, collecting raw data from devices separately and together as an aggregated power source. The data processing included a smoothing process and cleaning up the data to be used for disaggregation algorithms. Phase 2 is where the implementation of the algorithms occur and deep analysis on the 2 algorithms being used to execute the disaggregation process which is GSP and PrunedNILM.

3.3 Data sets and Requirements

Disaggregation of Energy can be a difficult problem for homes when the list of home appliances is unknown and unfamiliar. Therefore it would be a good idea to verify ahead of time whether or not a home appliance is suitable for Energy Disaggregation. Previous studies as in [18] have also shown heavy importance to the data requirements and typical characteristics a dataset should possess. A list of requirements is also included which consists of Reproduciblity, Multiplicity, Similarity And Inconsistency. A common requirement would be for the device to have the same or almost identical usage pattern every time the device is run on the same mode, if there is a new pattern every time the device is turned on, it would not be suitable to be part of the aggregated data and to then perform energy disaggregation.

3.4 Pruning

Machine learning software optimisation can take many different forms, Pruning is one of the more common ones.



(a) Before Pruning (b) After Pruning

Figure 1: Diagram of how pruning works

As shown in 1, the essence of pruning refers to removing extraneous values from the weight tensors. In order to eliminate what we isbelieved to be pointless connections between the layers of a neural network, we are essentially setting the values of the neural network parameters to zero. To help the neural network adjust to the changes,

this is done during training.

There are 4 pruning techniques that are investigated in this paper: Entropy-Based Pruning(EBP) [14], Relative Threshold Pruning(RTP) [2], Structured probabilistic pruning(SPP) [34], and constant sparsity Low Magnitude Pruning [1]. These pruning strategies operate by removing any weights from the networks that do not significantly contribute to the output of the networks and do not impair the algorithm's performance. Additionally, by lowering the number of CNN filters and neurones, the architecture of seq2point learning might be made lighter. The methods are further discussed and evaluated in the Artefact Development and Evaluation sections.

4 Artefact Development Approach

The goal of Energy Disaggregation in this project is to separate each appliance's power consumption from an overall power consumption signal. Lets assume there are N different appliances connected to 1 smart plug, the equation of the aggregated consumption would be represented like this:

$$x_t = \sum_{i=1}^{N} (y)_t^i$$

where

$$(y)_t^i$$

is denoted as the power consumption of appliance i at time t.

4.1 Data Collection Application

To collect data, firstly a smart plug is required. this project uses an Athom smart plug provided from the team, the first step is to set it up with your WiFi. The plug can be connected by looking into the devices connected to the WiFi and retrieving the URL that the smart plug is sending the collected data to in intervals.Once the smart plug is connected to the wifi, the router acts as the gateway where it is uploaded to the cloud. The data uploaded to the cloud is in raw json text, to convert this, a data collection application is created that collects data directly from the specified URL by sending get requests with a sample rate of 1 second, writes and saves this data which is in power(watts) and current(amps) against a UNIX timestamp into a csv file and saves it according to the users arguments.Data collection needs to be done while the program is running so that the data can be written and saved while a device is connected to the smart plug.

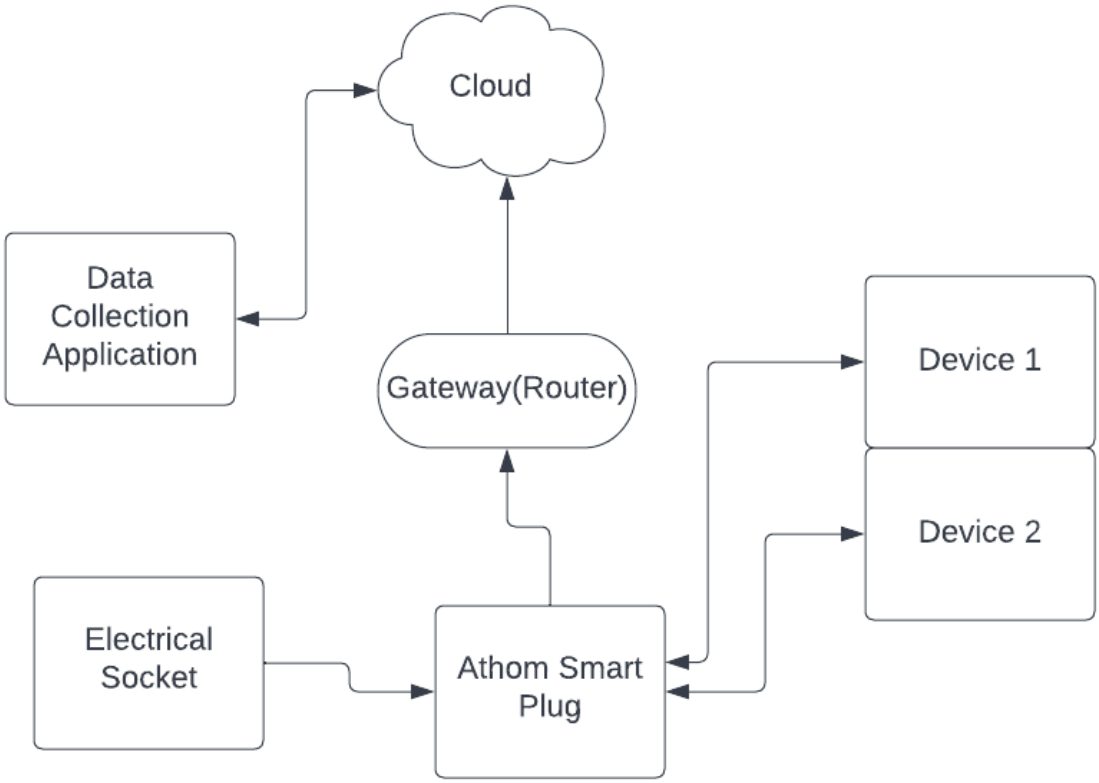


Figure 2: Flowchart of data collection process

Figure 2 shows how data is collected. Once the smart plug is connected to an electrical socket, for the initial setup you would have to manually connect it to the WiFi and discover the URL that data is being uploaded to, after which the router will start acting as the gateway to send the data collected to the cloud. The data is uploaded at a sample rate of 6-9 seconds, however my data collection application requests, and writes data every second. Once the application is stopped, it saves all the data and the csv file can then be used to analyse and evaluate the data for trends or any other relevant observations. This smoothed data is then used in the Graph Signal Processing method.

For PrunedNILM, the dataset that is being used for evaluation is a publicly available data-set, The UK REFIT Electrical Load Dataset[24] contains timestamped data sampled at 8 seconds intervals in Watts for 20 households at the appliance and aggregate level over a period of almost 2 years (2013-2015). The dataset is substantially large and contains 6GB worth of power readings which would be suitable for creating models that are broadly applicable to the appliance domain. Two

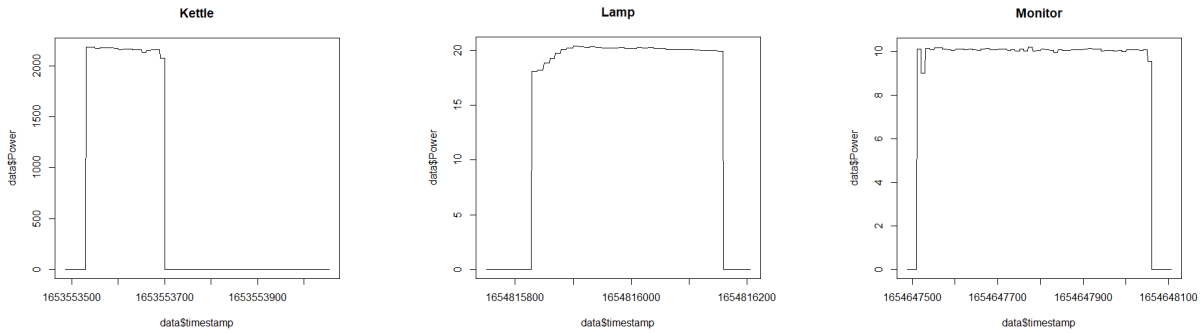
appliances will be employed because training a model for every target appliance would need a lot of processing power and time.

4.2 Phase 1

Phase 1 of this research mainly involves discovering the state of the art, collecting lots of data, analysing, and finally smoothing the data which is later evaluated to find what would be most suited to the solve the problem of disambiguating aggregated power data.

Discovering state of the art involves all the research and reading done within the field of Energy disambiguation, finding different methods and ideas on what could be the approach and research direction needed to solve this problem.

A few examples of individual devices power consumption and aggregated power consumption graphs collected with the application are displayed below:



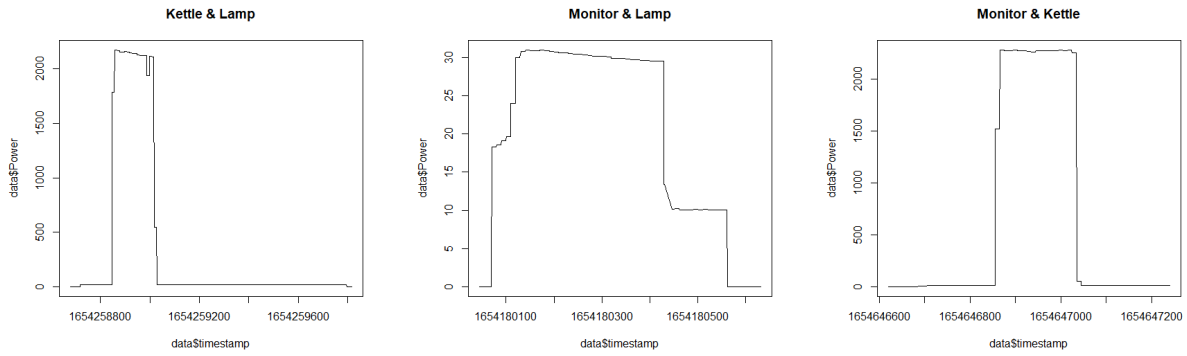
(a) Kettle power consumption

(b) Lamp power consumption

(c) Monitor power consumption

Figure 3: Individual power usage graphs

Figure 3 are the individual power consumption data of a Kettle, Monitor and Lamp collected from the data collection application. The x axis indicates the time in the UNIX timestamp, and the y axis displays the power in Watts. The 3 appliances chosen to be displayed is because they have clear trends that would give a better illustration for pattern recognition.



(a) Kettle & Lamp power consumption

(b) Monitor & Lamp power consumption

(c) Monitor & Kettle power consumption

Figure 4: Aggregated power usage graphs

Figures 4 shows the aggregated power consumption between the 3 devices. Even without any analysis done, the trends are clear enough to notice that the start and end points of the kettle is very distinct due to the huge amount of energy the kettle takes to heat up, where as with the monitor and lamp, it can be seen that at the beginning the lamp was switched on and after a few minutes the monitor was turned on, before it ends the lamp is switched off and then its only the monitor which explains the sudden drop and then steady power consumption. This can be seen when the 2 individual devices are grouped together in Figure5

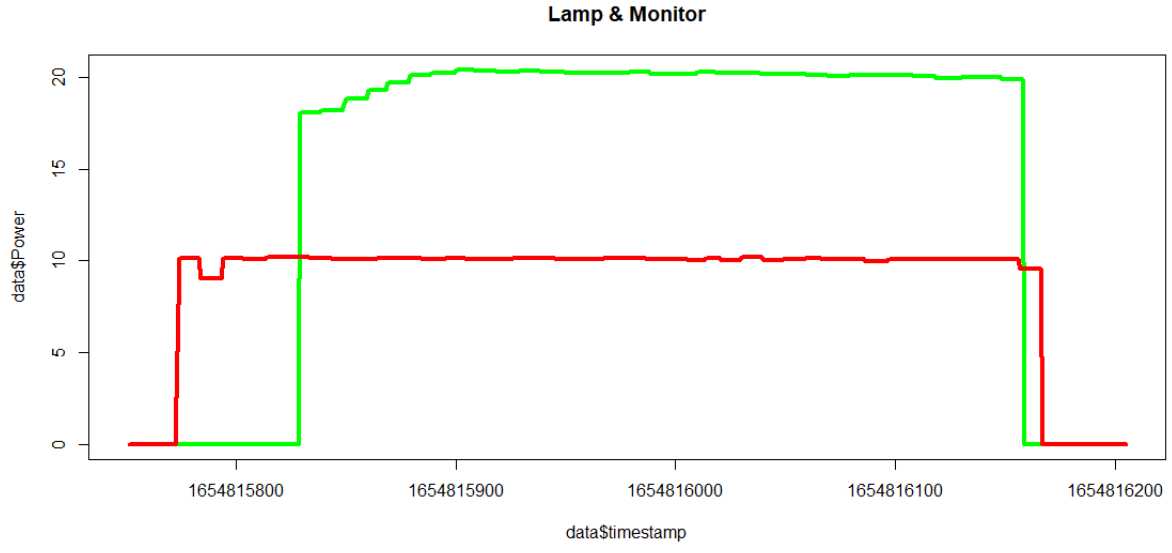


Figure 5: Individual power usage of monitor and lamp grouped together

Figure 5 shows the combined graphs of the individually collected power usage of the Monitor and Lamp. The green line is the power consumption from the Lamp, and the

red line represents the power consumption from the Monitor. The trends of these 2 appliances are similar with sudden spikes but the lamp having a slight gradual increase when switched on and both suddenly dropping to 0 when switched off.

4.2.1 Smoothing

The purpose of smoothing for this particular sets of data is to clean out the noise and get a smooth graph so that when a pattern recognition algorithm is run through, it would be able to catch the trends and the noisy data would not interrupt the algorithm.There are many techniques to reduce the noise like weighted moving average, kernel smoother, double exponential smoothing but the 3 different methods of smoothing that are chosen to evaluate for the time series data are, the first one being SMA which stands for Simple Moving Average, second one being Gaussian smoothing, and lastly the LOESS smoothing method.

Simple Moving averaging techniques are a straightforward way to smooth out demand data from the past. Almost all time series approaches are built on the foundation of these decomposition components.Moving averaging works on the assumption that demand measurements (seconds/minutes/days) that are close in time are likely to have comparable values.The simple moving average uses a average of the past k observations to create a future one-period-ahead forecast. It means that all k data points have the same weights. [16] The formula for a simple moving average can be given as:

$$y(n) = 1/N \sum_{k=0}^{N-1} x(n-k)$$

Gaussian smoothing is a kernel smoother that works by averaging nearby points to shape a defined Gaussian curve, which looks like a bell curve.[26] This method is used to clean out noise and cut out the sharp edges.The formula for a Gaussian kernel can be given by:

$$K(x^*, x_i = exp(-(x^* - x_i)^2/2b^2)$$

Locally Weighted Polynomial Regression Method often referred to as LOESS smoothing. Unlike conventional smoothing methods LOESS uses a constant smoothing

factor all the way to 0 axis, Another advantage is it allows non integer smoothing factor values to be specified, which allows finer control over the degree of smoothing. Adding to that it can also perform automated extrapolation across gaps or undefined regions which would remove outlying data and give out smoother data. The degree of smoothing is specified in the LOESS algorithm by a factor: α

4.3 Phase 2

Phase 2 of this project is where the pattern recognition and the actual disaggregation of Energy data takes place. There are 2 distinct approaches, GSP and PrunedNILM which is evaluated and analysed separately. The reason for using different datasets is due to the fact that GSP does not need a huge chunk of data, which is why the data being used is the one that was collected. However to perform a machine learning algorithm, huge amount of data is required in order to train and test the data, hence a public dataset is used.

4.3.1 Graph Signal Processing

A popular approach to energy disaggregation is using one of the of Non-Intrusive Application Load Monitoring (NALM)approaches, supervised or unsupervised, but require training to build appliance models, and are sensitive to appliance changes in the house, thus requiring regular re-training. This problem is addressed by graph signal processing by suggesting a NALM strategy that doesn't need any special training. The key concept is to do adaptive signal processing on the growing field of graph pattern matching, signal grouping, and thresholding. The performance limits of this approach is determined and further demonstrates its usefulness in practice.

GSP is based on graph signals, which are found by indexing a dataset by a graph's nodes. The primary concept is to visualise a dataset using a graph made up of a number of nodes and a weighted adjacency chart, each of the graph's nodes represent an element in the dataset. The adjacency matrix describes all edges in the graph and their weights, where the weights provided to the edges show how similar or correlated the nodes are to one another.

The suggested approach disaggregates any aggregate active power dataset without any

prior information, including knowledge of appliances contributing to the aggregate or their number. It is based on graph signal processing (GSP), a new discipline where a dataset is represented by a discrete signal that is indexed by nodes in a graph. By embedding the structure of signals onto a graph, GSP provides an alternative to traditional approaches to signal processing. This results in a powerful, scalable, and adaptable technique ideal for a variety of applications.

GSP is utilised three times for the disaggregation of an active power signal: once for robust event detection, once for clustering, and once for feature matching. The approach is event-based and relies only on time-series data without any training.The process relies on sliding time windows with adjustable duration of window sizes ranging from a month, week, down to a day and an hour have been tested, However shorter windows are also an option. Disaggregated appliances are named once each window has been processed using an existing database, whose contents are regularly updated. In the event that an appliance is not found, fresh signatures using graph signal processing are used and would then be added as a new signature and the appliance would be added with an arbitrary label until it is later updated with a suitable label name.

Event based approaches first pinpoint windows of events that has statistically significant shifts in active power that would suggest one or more appliances operational condition have changed state (On/Off). Fixed or adaptive thresholding is used to detect events, and the threshold must be high enough to exclude fluctuations in power from the same appliance yet low enough to detect low load. The most common method of event detection is edge detection, after events have been identified they are categorised into pre-established classes using a model during training.

Through the use of regularisation on the underlying graph, clustering is carried out in two levels, with each level forming a cluster from all data samples that are highly correlated to the first data sample. In level 1 we use the initial threshold to filter out low-magnitude edges, while in level 2 some of the clusters are modified by adjusting the threshold to the data samples. Each cluster after level 1 will have just pure positive or only pure negative edges. Cluster elements must be packed closely together since low cluster means will make samples more susceptible to noise. The cluster with a greater mean would therefore have superior quality for two clusters with the same variance. Stage 2 involves repeating the previous GSP-based clustering cycles, but only for the low-quality clusters.

For the final feature matching stage, each ”positive” cluster with the ”negative” cluster

that has the closest absolute mean value because the final clusters comprise an equal number of ”positive” clusters (clusters with increasing power edges) and ”negative” clusters (clusters with decreasing power edges). Then for each positive-negative edged pair, GSP is performed to match the increasing power edge with a decreasing power edge, by utilising magnitude differences as well as time intervals between the edges as two matching features.

Following the formation of the positive and negative pairings through feature matching, each pair represents a potential appliance state. The matched samples from the two paired clusters identify the start and end of the appliance operating event. The disaggregated signature is then compared to a database of appliance signatures that are already accessible for that particular household to label each disaggregated event. If the database is empty or the appliance is missing from the database, does not find a match, appliances would be added in the same way as they are disaggregated with an arbitrary label, up until the user checks their label and then manually changes inputs a label name.

4.3.2 Pruned-NILM

The common NILM procedure uses a series of aggregate power data To map a target appliance's power consumption, these learning algorithms need a lot of time and hardware.Domestic energy disaggregation technologies for house smart metres are now mostly unavailable due to this restriction. For instance, there are over thirty million weights in the seq2point learning model. To deliver a mobile energy disaggregation system, methods must be used to reduce this number as much as possible. The research suggests to examine pruning the network to reduce both the size and inference time of such neural network methods in order to mitigate this restriction. There are 4 pruning methods investigated to shrink the data, entropy-based pruning [14], relative threshold pruning [2], Structured probabilistic pruning [34], and constant sparsity low magnitude pruning [1]. Additionally, by lowering the number of CNN filters and neurones, the architecture of seq2point learning might be made lighter. The machine learning algorithm used is sequence-to-point with a simplified architecture of the REFIT data set.

Sequence to Point machine learning method trains a neural network solely by

predicting the midpoint element rather than training the network to predict a window of appliance readings. The concept is basically the mains window serves as the network input and the midpoint of the corresponding window of the target appliance serves as the network output. This approach assumes that the mains window's non-linear regression is used to represent the midway element. This assumption is based on the argument that we predict the status of the appliance's midpoint component to be related to the data of the mains before and after that midpoint.

Network weight pruning involves reducing the amount of weights while maintaining performance in order to lower the size of the network. A brief description of the 4 pruning methods that are used in PrunedNILM:

- Entropy-based Pruning: An entropy based approach uses the information contribution of a weight to assess if it needs be pruned. It aims to lessen the decrease in inference accuracy that results from network sparsification. The procedure layers the network, assuming that each layer's weight is a Gaussian Distribution. Recursive steps are taken until the training has been completed.[14]
- Relative Threshold Pruning: The Relative Threshold Pruning (RTP) method is implemented to identify and enforce a threshold value below which pruning takes place. The most effective algorithm on display chooses pruning criteria based on the size of each layer. As a result, the distribution of a layer's weights becomes the only factor that influences how many weights are trimmed. To guarantee that models may still converge, the thresholds for convolutional layers must have a smaller range and must have unique values for their upper and lower thresholds.[2]
- Structured Probabilistic Pruning: The structured probabilistic pruning (SPP) algorithm groups weights by filters and prunes the whole network at the same time, in contrast to other pruning algorithms where pruning is done layer by layer. Additionally, 2D convolutional layers like those seen in the seq2point model are less likely to be substantially pruned comparatively with full connected layers. This way it is possible to manage the time complexity even when the network is deeper.[34]
- Constant Sparsity Low Magnitude Pruning : A straightforward low magnitude trimming approach is provided by TensorFlow's Model Optimisation library. During training, this approach subtracts from a neural network model a specific, predetermined percentage of the weights with the lowest absolute values.

Tensorflow includes 2 pruning toolkits,first one known as polynomial decay low-magnitude pruning pruning which gradually reduces the number of weights until the target sparsity level is reached. The second kit is called constant sparsity low-magnitude pruning which generates a model with the desired sparsity immediately and makes adjustments to the weights that were eliminated to improve precision at each pruning iteration. [1]

5 Results and Evaluation

This section includes the evaluation and results of the project. The contents are divided into 2 phases for more clear evaluation. The first phase includes data preparation and processing where different smoothing methods were analysed and the optimum one was chosen to be implemented. Phase 2 of this section includes the energy disaggregation methods and is further divided into 2 sections for GSP and PrunedNILM where results are displayed and evaluated.

5.1 Phase 1

The dataset that was chosen to analyse, evaluate and test the smoothing methods is the laptop charger power consumption. From the observations of collecting power consumption of devices around the household, the trend of power consumption in the laptop charger was the most noisy which is perfect to display the smoothing analysis. The testing process involved 3 distinct smoothing methods - SMA, Gaussian kernel and LOESS that is elaborated in 4. To test these smoothing methods a program was created in R studio with R as the programming language.

The original dataset of the laptop charger along with the 3 smoothing methods are displayed below:

As shown in figure 6 the x axis showing the timestamp in a UNIX format and the y axis showing the Power in watts(W). The graph is noisy and almost impossible to run a pattern recognition technique without getting a completely new and haphazard graph every time the laptop is being charged.

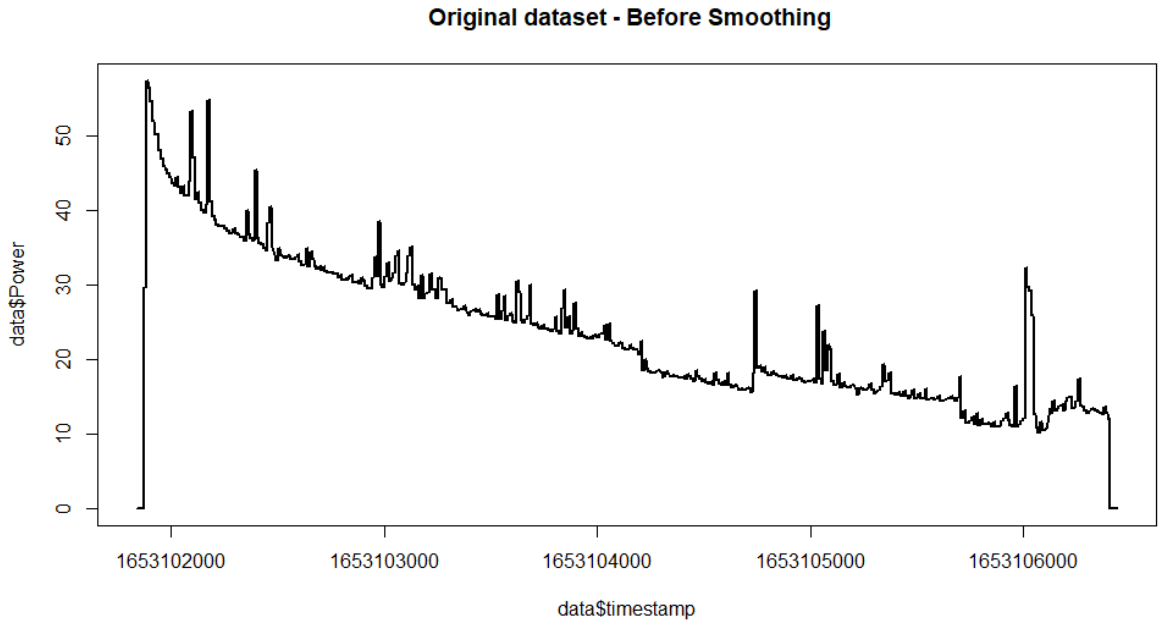


Figure 6: Original dataset - Before Smoothing

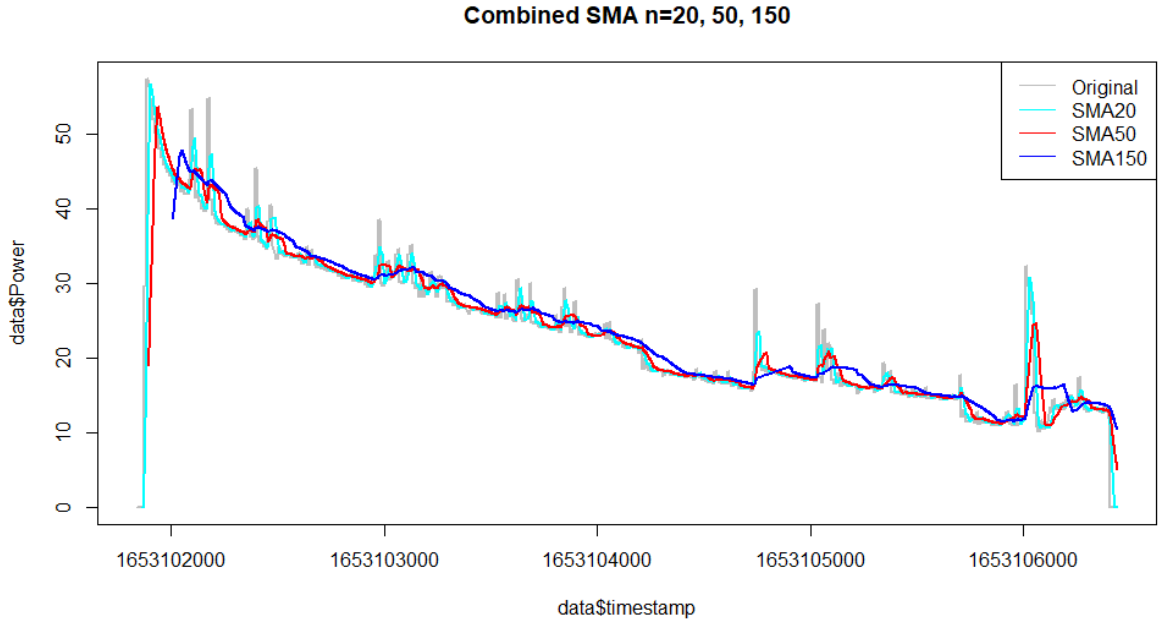


Figure 7: After Smoothing with SMA

In figure 7 displays the performance of Simple Moving Average where n is the number of periods taken after a certain point. SMA(n=20) was good as it cleaned out the data, SMA(n=50) was a bit better with cleaning but resulted in some awkward bumps which may alter the original pattern. SMA(n=150) was too much smoothing and pretty much

changed the graph so would not be ideal for this scenario.

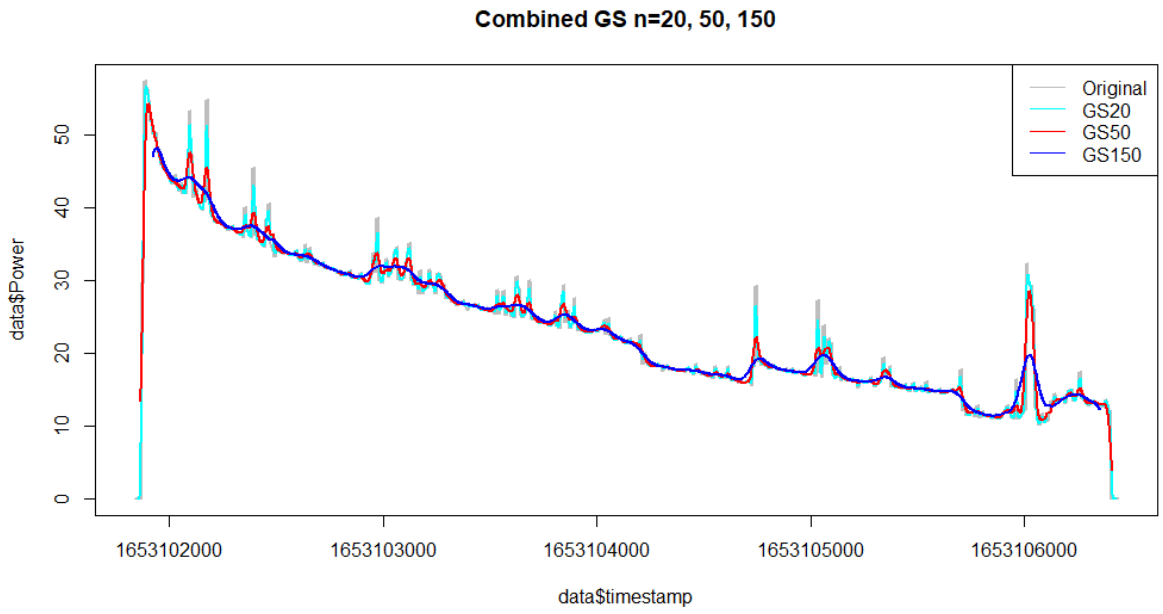


Figure 8: After Smoothing with Gaussian Kernel

In figure 8 displays the performance of Gaussian Kernel Smoothing, the results were pretty similar to the ones used in SMA such as GS20 was pretty good , GS50 was better with cleaning as it smoothed out the data, while also staying not far from the original data. GS150 like SMA150 was too overly smoothed and hence gets disregarded.

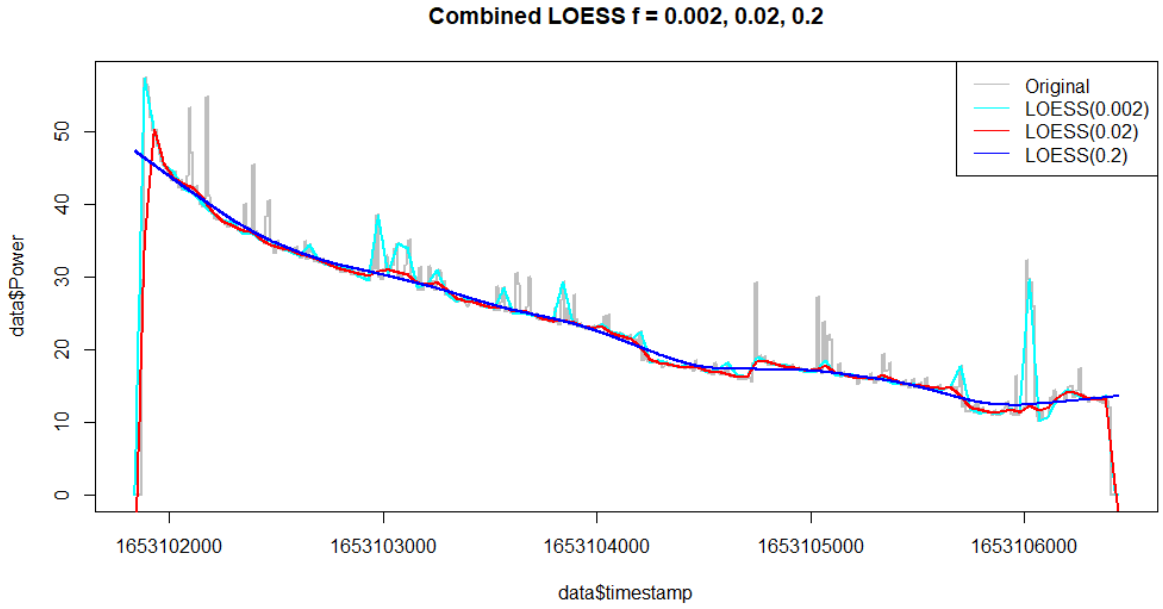


Figure 9: After Smoothing with LOESS

In figure 9displays the performance of the final LOESS(Locally Weighted Polynomial Regression Method) Smoothing, The results were different from the other 2 methods, with f being the function to increase or decrease the value used for the smoother span. The larger the number given to the function, the smoother the graph would turn out. LOESS(0.002) shows a pretty neat graph with the data being smoothed out and key values seem to still be visible, However LOESS(0.02) and LOESS(0.2) both have too much smoothing resulting in an almost straight line graph which defeats the purpose of smoothing for this project.

According to our requirement which is simply to get a more smoother graph to make it easier for pattern recognition algorithms to recognise the specific device, the best 3 algorithms would be the SMA20, GS50 and LOESS(0.002) from the 3 tested methods. As observed in Figure 10 all 3 would suit the criteria, However GS(50) stands out since its got a smooth graph as well as key spikes and drops indicating that it is no too far from the original, while also showing a clear trend.

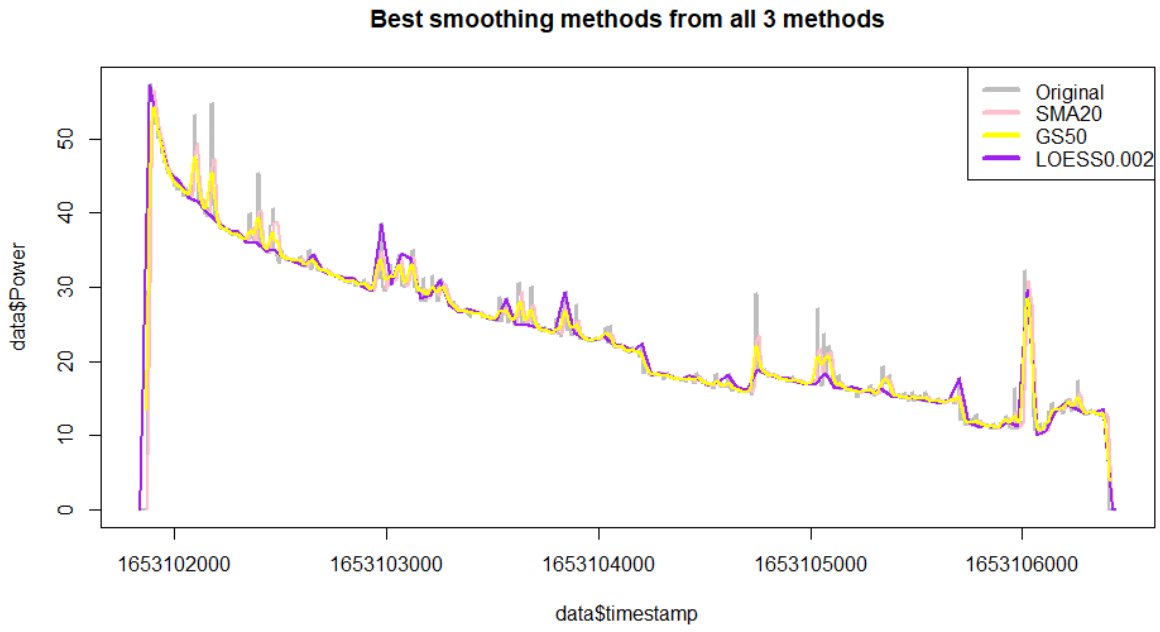


Figure 10: Best smoothing algorithms from each of the 3 methods

5.2 Phase 2

The section is further divided into GSP and Pruned-NILM and evaluation will be done separately for each algorithm along with the performance matrix. The basis on why the 2 algorithms can not be impartially compared is due to GSP being solely a pattern matching algorithm that uses a small-scaled dataset, where as PrunedNILM is a machine learning algorithm and needs a large sized dataset in order to be able to train and test data.

5.2.1 Graph Signal Processing

GSP starts with clustering of the positive and negative edges, each power edge corresponds to one node in the GSP cluster. All edges which are similar in magnitude are grouped into a cluster, this process continues until every event is clustered. The next step is to pair up each positive cluster with the nearby negative cluster in terms of magnitude. Then, using feature matching, we couple each rising edge with an ideal falling edge for each pair of positive and negative clusters. Each disaggregated event is labelled by comparing the disaggregated signature after the algorithm is run, with an

existing database of appliance signatures available for that particular household

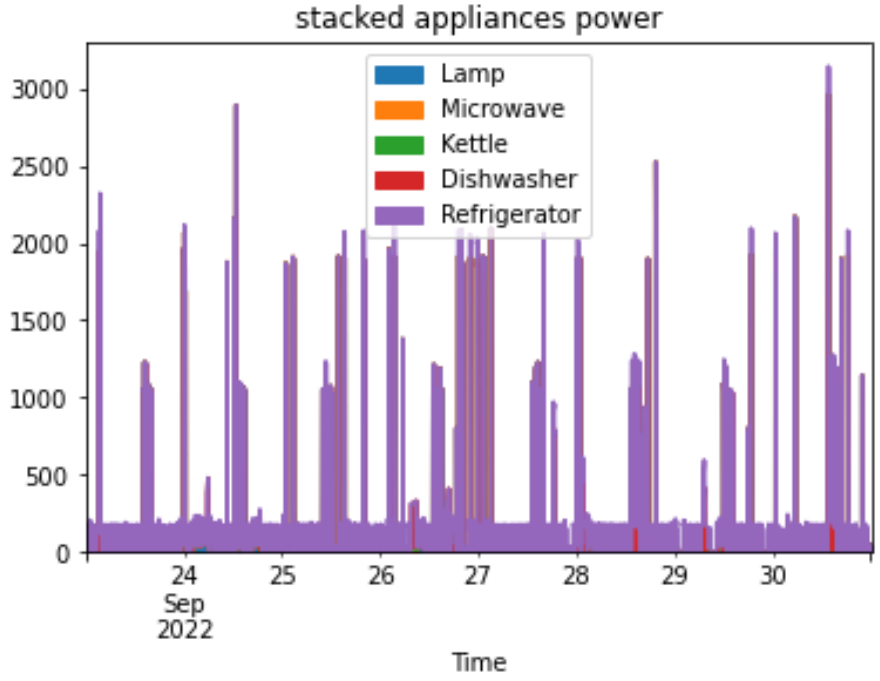


Figure 11: Stacked appliances aggregated power for 1 week

Figures 12 and 11 show the aggregated combined power and the individual power consumption of each of the 5 appliances recorded.

Figure 13 shows the pie charts and the comparisons between the actual power consumption from signatures against the disaggregated consumption's after GSP is performed. The disaggregation error between the algorithm and the ground truth signatures is 5.3% , the unidentified section in the disaggregated power pie chart is the difference between the total loads of the individual, sub-metered appliances and the measured aggregate consumption.

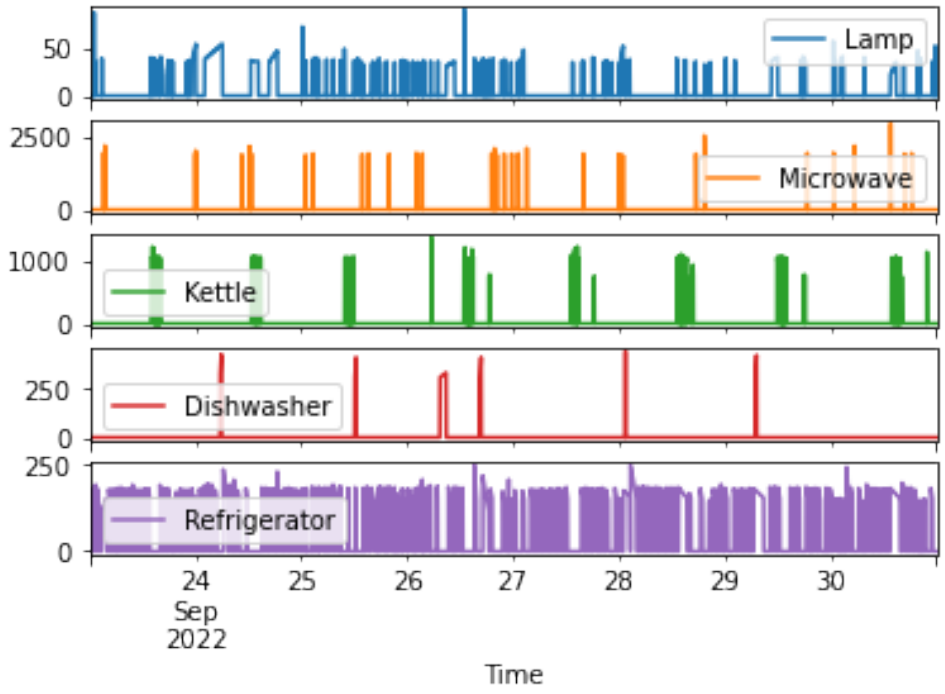
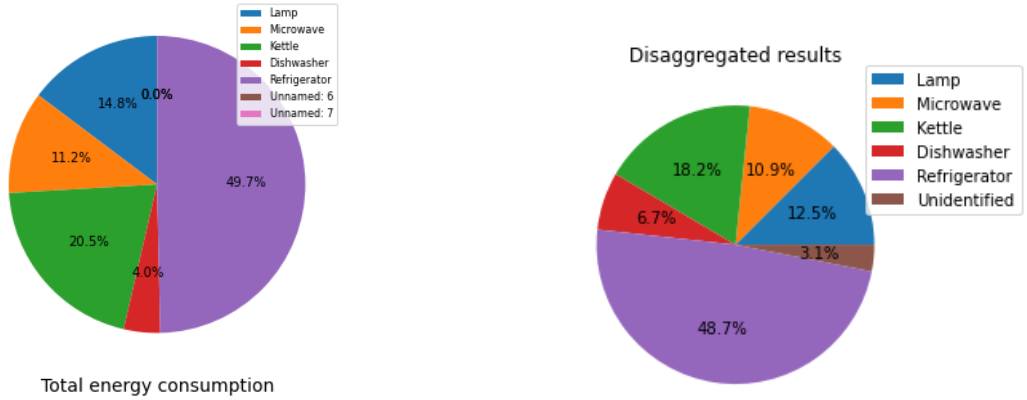


Figure 12: Individual appliances powers for 1 week



(a) Aggregated power consumption (b) Dis-aggregated individual consumption

Figure 13: Individual power usage graphs

5.2.2 Pruned-NILM

In order to evaluate the pruning algorithms, training the data is required and an appliance needs to be chosen. For this evaluation, the kettle is chosen in order to run through all 4 pruning algorithms and choose the best out of the lot for sequence-to-point learning. The dataset generator took 984.2 seconds to train a dataset that contains 60,429,234 rows of data from the REFIT database with the kettle being the

appliance selected.

The error metrics used to evaluate the performance of the models are Mean Absolute Error (MAE) and Mean Square Error and Mean Squared Error(MSE)

The total number of trainable weights: 30,708,249

	EBP	RTP	SPP	LMP
No. of weights	21,251,411	18,479,487	30,695,431	9,239,743

Table 1: The 4 pruning methods applied to the Sequential-to-point Model: Entropy-Based Pruning (EBP), Relative Threshold Pruning (RTP), Structured Probabilistic Pruning (SPP), Low Magnitude Pruning (LMP)

Mean Square Error (MSE)

Appliance	EBP	RTP	SPP	LMP
Kettle	0.1153	0.1272	0.0993	0.1105

Table 2: Mean Square Error for each of the pruning methods with the Kettle

Mean Absolute Error (MAE)

Appliance	EBP	RTP	SPP	LMP
Kettle	0.0768	0.08214	0.0644	0.0728

Table 3: Mean Absolute Error for each of the pruning methods with the Kettle

The mean square errors and mean absolute errors were calculated and displayed in 2 and 3 respectively. According to the results, the Structured Probabilistic Pruning (SPP) algorithm produced the best model with the lowest error values when compared to the other models. The total sparsity ratio of the model was minimal even though SPP eliminated 24% of the weights from the convolutional layers since SPP removes weights only from the convolutional layers of a model.

The next best performing model was Low Magnitude Pruning (LMP) algorithm. Additionally, LMP has the largest sparsity ratio (70%) of any kettle model, which means that the model's total number of non-zero weights was decreased from 30 million to 9.2 million. This fairly considerable reduction appears to offer the best balance between the sparsity ratio and error with little to zero performance deterioration.

Entropy-Based Pruning (EBP) errors were similar to those of LMP. However, the sparsity ratio of the model after EBP pruning was 39% lower than that of the LMP model. When compared to the other models, EBP underwent a large amount of pruning while keeping strong inference capabilities.

The least impressive model was the Relative Threshold Pruning (RTP) algorithm. the significantly higher error values suggest that the algorithm is not well-suited for pruning sequence-2-point learning energy disaggregation.

5.2.3 Performance matrix

Evaluation in performance and accuracy is important in order to assess the success of the suggested algorithm. Hence the performance matrix used to find the accuracy of the algorithm's capacity to determine if a device is on or off. [19]. Accuracy and Estimated Accuracy can be defined as:

$$Accuracy = \frac{Correct!matches}{Total!possible!matches}$$

$$Est.Acc. = 1 - \frac{\sum_t \sum_m |P(hat)^t_m - P^t_m|}{2. \sum_t \sum_m P^t_m}$$

where t is the number of test samples, m is the appliance, P(hat)mt is referred to as the estimated power of appliance m at instance t, (P)mt is the actual power consumption.

In order to find the precision of the algorithm, the metrics used is Precision(PR), Recall(RE) and F-Measure(FM) adapted from [19]. The 4 values given for each appliance are True Positive(TP), False Positive(FP), True Negative(TN) and False Negative(FN).

$$Precision = TP/(TP + FP)$$

$$Recall = TP/(TP + TN + FN)$$

$$F - Measure = 2 * (PR * RE/(PR + RE)$$

Appliance	TP	FP	TN	FN	PR	RE	FM
Kettle	37	3	0	2	0.93	0.95	0.94
Lamp	16	7	1	3	0.84	0.8	0.82
Monitor	15	7	2	2	0.75	0.79	0.77
Laptop	29	27	7	11	0.59	0.62	0.6

Table 4: Performance of the approach with data collected

Table 4 displays the performance from the collected data, relatively laptop has a low F-Measure for which the culprit is likely due to the randomness in power usage at different power levels when the laptop is charged. However the kettle shows to have a high F-Measure since the power usage is dis-similar to any of the other device signatures, unlike the monitor and the lamp which have similar signatures which explains the false positive readings.

Appliance	TP	FP	TN	FN	PR	RE	FM
Microwave	12	0	8	4	1.00	0.50	0.67
Kettle	43	3	7	0	0.93	0.86	0.90
Freezer	51	88	15	20	0.37	0.59	0.45
Refrigerator	22	84	12	1	0.21	0.63	0.31
TV	9	5	2	3	0.64	0.64	0.64
Washing Machine	7	4	0	1	0.64	0.88	0.74

Table 5: Performance of the approach with a single house power usage from the REFIT data-set

Table 5 displays the performance from the REFIT dataset. Note that the readings are only from the known devices since the data contains a few unknown devices which were left out for this performance matrix. The Refrigerator showed a poor performance in the F-measure due to the similar power usage with the particularly the freezer, the kettle once again has a good F-measure score mainly because of the unique power usage, with very high power at a short period of time unlike many other appliance readings that were taken.

6 Conclusion & Future Work

The research demonstrates sufficient literature of the existing methods used to disaggregate Energy consumption from aggregated sources. The data requirements, the data collection process and a detailed smoothing analysis is also listed to successfully carry out a disaggregation algorithm. There are 2 distinct algorithms carried out within this paper: GSP and PrunedNILM, in which the first one uses low-volume data sets and does not involve any training or testing, it is an unsupervised low-rate NALM approach that can accurately capture signal patters and has low implementation complexity. The latter is a more traditional NILM approach but just pruned in order to be able to lighten the weight of the datasets needed for training and testing. GSP showed that the error percentage between the ground truth signatures and disaggregated solution was 5.3%. PrunedNILM showed to have had the best pruning results with Structured Probabilistic Pruning (SPP) with MSE as low as 0.0993 and MAE 0.0644 which was the lowest errors compared to the other pruning models. Low Magnitude Pruning was a close second with also low error values for MSE and MAE, the results were 0.1105 and 0.0728 respectively. Results and analysis clearly proved that both methods are well constructed and are suitable for energy dis-aggregation with both short term low volume data as well as long term high volume data with different algorithms which can be chosen according to your dataset and requirements. GSP is much faster and involved no training or testing at all and comparatively less complex, whereas Pruned-NILM can handle extremely large datasets and uses sequence-to-point and NILM toolkit which is a proven machine learning algorithm that has been in use for many years.

6.1 Future Work

Just like with anything in the technology world, there is always room for further improvement and advancing the technology. There is a lot of potential in the field of energy disambiguation especially with the recent awareness of climate change and the hike in energy prices, more people are interested in the statistics of their energy usage and in turn would allow more individuals to help reduce energy wastage. There are various research papers done on this field but there are still gaps that need to be filled like optimising already available algorithms to make it faster and reduce the complexity.

References

[1] Tensorflow model optimization toolkit — pruning api, (2019).

[2] A. H. Ashouri, T. S. Abdelrahman, and A. Dos Remedios, Retraining-free methods for fast on-the-fly pruning of convolutional neural networks, Neurocomput., 370 (2019), p. 56–69.

[3] J. Barber, H. Cuayáhuitl, M. Zhong, and W. Luan, Lightweight non-intrusive load monitoring employing pruned sequence-to-point learning, in Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring, NILM'20, New York, NY, USA, 2020, Association for Computing Machinery, p. 11–15.

[4] N. Batra, J. Kelly, O. Parson, H. Dutta, W. Knottenbelt, A. Rogers, A. Singh, and M. Srivastava, NILMTK, in Proceedings of the 5th international conference on Future energy systems, ACM, jun 2014.

[5] N. Batra, R. Kukunuri, A. Pandey, R. Malakar, R. Kumar, O. Krystalakos, M. Zhong, P. Meira, and O. Parson, Towards reproducible state-of-the-art energy disaggregation, BuildSys '19, New York, NY, USA, 2019, Association for Computing Machinery, p. 193–202.

[6] N. Batra, A. Singh, and K. Whitehouse, If you measure it, can you improve it? exploring the value of energy disaggregation, in Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments, BuildSys '15, New York, NY, USA, 2015, Association for Computing Machinery, p. 191–200.

[7] M. Berges, E. Goldman, H. Matthews, and L. Soibelman, Learning systems for electric consumption of buildings, vol. 346, 08 2009.

[8] M.-U.-S. Chowdhury, A. Tahmid, M. A. Azmain, M. S. Chowdhury, and M. Hossam-E-Haider, Exponential smoothing technique in filtration of distorted radar signal, in 2022 International Conference for Advancement in Technology (ICONAT), 2022, pp. 1–5.

[9] E. Elhamifar and S. Sastry, Energy disaggregation via learning powerlets and sparse coding, Proceedings of the AAAI Conference on Artificial Intelligence, 29 (2015).

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

[11] G. Hart, Nonintrusive appliance load monitoring, Proceedings of the IEEE, 80 (1992), pp. 1870–1891.

[12] L. Hattam and D. V. Greetham, Energy disaggregation for smes using recurrence quantification analysis, in Proceedings of the Ninth International Conference on

Future Energy Systems, e-Energy '18, New York, NY, USA, 2018, Association for Computing Machinery, p. 610–617.

[13] F. Hidiyanto and A. Halim, Knn methods with varied k, distance and training data to disaggregate nilm with similar load characteristic, APCORISE 2020, New York, NY, USA, 2020, Association for Computing Machinery, p. 93–99.

[14] C. Hur and S. Kang, Entropy-based pruning method for convolutional neural networks, J. Supercomput., 75 (2019), p. 2950–2963.

[15] S. Kaleg, A. Hapid, Amin, A. C. Budiman, and Sudirja, Data smoothing of a real-life driving test result, in 2020 International Conference on Sustainable Energy Engineering and Application (ICSEEA), 2020, pp. 1–6.

[16] S. A. Karim and S. A. Alwi, Electricity load forecasting in utp using moving averages and exponential smoothing techniques, Applied Mathematical Sciences, 7 (2013), pp. 4003–4014.

[17] J. Kelly and W. Knottenbelt, Neural nilm: Deep neural networks applied to energy disaggregation, BuildSys '15, New York, NY, USA, 2015, Association for Computing Machinery, p. 55–64.

[18] G. Kim and S. Park, A study on data requirements for power disaggregation, in 2020 3rd International Conference on Power and Energy Applications (ICPEA), 2020, pp. 141–144.

[19] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han, Unsupervised disaggregation of low frequency power measurements, vol. 11, 04 2011, pp. 747–758.

[20] J. Z. Kolter, S. Batra, and A. Y. Ng, Energy disaggregation via discriminative sparse coding, NIPS'10, Red Hook, NY, USA, 2010, Curran Associates Inc., p. 1153–1161.

[21] G. C. Koutitas and L. Tassiulas, Low cost disaggregation of smart meter sensor data, IEEE Sensors Journal, 16 (2016), pp. 1665–1673.

[22] H. Lange and M. Bergés, Bolt: Energy disaggregation by online binary matrix factorization of current waveforms, in Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments, BuildSys '16, New York, NY, USA, 2016, Association for Computing Machinery, p. 11–20.

[23] P. Mtshali and F. Khubia, A smart home energy management system using smart plugs, in 2019 Conference on Information Communications Technology and Society (ICTAS), 2019, pp. 1–5.

[24] S. L. . S. V. Murray D., An electrical load measurements dataset of united kingdom households from a two-year longitudinal study, (2017).

[25] S. Pandey and G. Karypis, Structured dictionary learning for energy disaggregation, in Proceedings of the Tenth ACM International Conference on Future Energy Systems, e-Energy '19, New York, NY, USA, 2019, Association for Computing Machinery, p. 24–34.

[26] S. Regmi, Gaussian smoothing in time series data, (2021).

[27] P. A. Schirmer, I. Mporas, and M. Paraskevas, Energy disaggregation using elastic matching algorithms, Entropy, 22 (2020).

[28] S. R. Shaw, C. Abler, R. F. Lepard, D. Luo, S. B. Leeb, and L. K. Norford, Instrumentation for high performance nonintrusive electrical load monitoring, 120 (1998), pp. 224–229.

[29] R. Sinha, S. Spoorthy, P. Khurana, and M. G. Chandra, Power system load data models and disaggregation based on sparse approximations, in 2016 IEEE 14th International Conference on Industrial Informatics (INDIN), 2016, pp. 292–299.

[30] F. Sultanem, Using appliance signatures for monitoring residential loads at meter panel level, IEEE Transactions on Power Delivery, 6 (1991), pp. 1380–1385.

[31] G. Tang, J. Chen, C. Chen, and K. Wu, Smart saver: A consumer-oriented web service for energy disaggregation, in 2014 IEEE International Conference on Data Mining Workshop, 2014, pp. 1235–1238.

[32] G. Tang, K. Wu, J. Lei, and J. Tang, Plug and play! a simple, universal model for energy disaggregation, 2014.

[33] S. Tomkins, J. Pujara, and L. Getoor, Disambiguating energy disaggregation: A collective probabilistic approach, in Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, 2017, pp. 2857–2863.

[34] H. Wang, Q. Zhang, Y. Wang, and H. Hu, Structured probabilistic pruning for convolutional neural network acceleration, in BMVC, 2018.

[35] C. Zhang, M. Zhong, Z. Wang, N. Goddard, and C. Sutton, Sequence-to-point learning with neural networks for non-intrusive load monitoring, in Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI'18/IAAI'18/EAAI'18, AAAI Press, 2018.

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