Non-intrusive load monitoring for low sampled aggregated data.

Thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science
in
Building Science and Engineering by Research

by

Ronak Aghera
20161002
ronak.aghera@research.iiit.ac.in

International Institute of Information Technology, Hyderabad
(Deemed to be University)
January, 2022
It is certified that the work contained in this thesis, titled “Non-intrusive load monitoring for low sampled aggregated data.” by Ronak Aghera, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Prof. Vishal Garg
To my Parents and Sister
Acknowledgments

I want to take this opportunity to express my sincere appreciation to everyone who not only helped me in the completion of my research thesis but also made my experience at IIIT Hyderabad an unforgettable one.

First and foremost, I am highly grateful to my supervisor Prof Vishal Garg for his invaluable advice, continuous support and encouragement, and patience during my Masters’s study. He gave me the freedom to choose my research area and explore the domain of Non-intrusive load monitoring. This thesis would not have been possible without his field experience, constant encouragement, valuable feedback, and insights into my work.

Center for IT in Building Science has become an integral part of my life in the last few years. I want to thank all my colleagues in this research center – Aviruch Bhatia, Sam Babu Godithi, Raghunath Reddy, Gopal Chittalia, Sahil Chilana, Madhur Garg, Rishika Agarwal, and Sraavani Gundepudi for their immense support. Special thanks to Mr. Ragunath Reddy and Madhur Garg; this study would not have been completed without their valuable help and technical feedback.

My parents and sister have been reliable pillars of support and inspire me at every step through their life journeys, and I can’t thank my family enough for all they’ve done for me. I hope to keep making them proud in the future.

Finally, life at IIIT-Hyderabad would not have been what it was without the constant support of my friends. Ankit, thanks for being the go-to person to trouble and poker partner. Himansh, thanks for being around as a parent and ever ready for anything, mainly sport, treks, and drinks. Maanik, thanks for being a geeky friend and bearing with all nuisance. Kartik, thanks for being a supportive friend and always lifting up the mood. Lovish, thanks for always being crazy. Abhishek, thanks for being a caring friend. Sahil, thanks for always being a partner in thrilling things. Thank you guys for being around and bearing with me all this year.

Support from Funding Agency:

The Department of Science and Technology, India (DST) and Engineering and Physical Science Research Council, UK (EPSRC) provided joint funding for work under the India-UK partnership grant no. EP/R008434/1 for Residential Building Energy Demand in India (RESIDE) project.
Abstract

Energy consumption of the residential sector is increasing year after year with economic development, urbanization, and improvement of people’s living standards. In order to achieve more efficient energy consumption, it is crucial to provide appliance-wise energy consumption feedback to the users to save energy. Appliance wise energy consumption feedback allows users to identify faulty appliances, reduce energy demand and identify unnecessary active devices. Individual appliance energy consumption is monitored by two approaches intrusive load monitoring and non-intrusive load monitoring. Intrusive load monitoring uses smart plugs and smart strips on each device to monitor their power consumption. Non-intrusive load monitoring (NILM) estimates the individual appliance power consumption from the main meter aggregated data without using additional sensors. NILM is a cost-effective approach for giving appliance-wise energy consumption feedback. Non-intrusive load monitoring (NILM) is a blind source separation problem that requires a system to estimate the electricity usage of individual appliances from the aggregated household energy consumption. In this thesis, we will discuss about NILM methods and algorithms on low sampled (1 min) aggregated data in detail and identify the research gaps. To addresses those research gaps, we propose a novel deep neural network-based approach for performing load disaggregation on low-frequency power data (1 min) obtained from residential households. We combine a series of one-dimensional Convolutional Neural Networks and Long Short Term Memory (1D CNN-LSTM) to extract features that can identify active appliances and retrieve their power consumption given the aggregated household power value. We took appliances which are multi-state appliances (washing machine, dishwasher, microwave, and refrigerator) to test the algorithm. The disaggregation performance of the algorithm is measured using six metrics and compared with five state-of-the-art algorithms. To explore how well the algorithms generalize to unseen houses, the performance of the algorithms was measured in two separate scenarios: one using test data from a house not seen during training and a second scenario using test data from houses that were seen during training. Our neural net achieves better F1 scores (across all four appliances) than state-of-the-art algorithms and generalizes well to unseen houses with a lower number of trainable parameters. The algorithm is designed for low-power offline devices. Empirical calculations show that our model outperforms the state-of-the-art on Reference Energy Disaggregation Dataset (REDD) and UK-dale dataset. We have collected the electricity consumption data of 11 houses in India for 19 days as part of the Residential Building Energy Demand in India (RESIDE) Project. The proposed model on RESIDE dataset achieved better F1 scores and generalised well on the unseen houses.
## Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Motivation</td>
<td>3</td>
</tr>
<tr>
<td>1.3 Problem Statement</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Contribution of this thesis</td>
<td>4</td>
</tr>
<tr>
<td>1.5 Outline of the thesis</td>
<td>5</td>
</tr>
<tr>
<td>2 Literature review on NILM</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Appliance Level Feedback</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Non-Intrusive Load Monitoring (NILM)</td>
<td>9</td>
</tr>
<tr>
<td>2.2.1 Data Collection</td>
<td>11</td>
</tr>
<tr>
<td>2.2.2 Feature Extraction:</td>
<td>12</td>
</tr>
<tr>
<td>2.2.3 Appliance Identification and Power Retrieval:</td>
<td>16</td>
</tr>
<tr>
<td>2.2.4 Research Gaps</td>
<td>22</td>
</tr>
<tr>
<td>2.3 Deep Learning</td>
<td>23</td>
</tr>
<tr>
<td>2.3.1 Artificial Neural Network</td>
<td>24</td>
</tr>
<tr>
<td>2.3.2 Convolution Neural Network</td>
<td>28</td>
</tr>
<tr>
<td>2.3.3 Recurrent Neural Network</td>
<td>29</td>
</tr>
<tr>
<td>3 Proposed Methodology and Experiment</td>
<td>30</td>
</tr>
<tr>
<td>3.1 Proposed Methodology</td>
<td>30</td>
</tr>
<tr>
<td>3.1.1 Appliance Signature Extraction</td>
<td>31</td>
</tr>
<tr>
<td>3.1.2 Appliance Operating State Identification</td>
<td>32</td>
</tr>
<tr>
<td>3.1.3 Appliance Power Retrieval</td>
<td>33</td>
</tr>
<tr>
<td>3.2 Experiments</td>
<td>34</td>
</tr>
<tr>
<td>4 Experimental Setup</td>
<td>38</td>
</tr>
<tr>
<td>4.1 Datasets</td>
<td>38</td>
</tr>
<tr>
<td>4.2 Data Pre-processing and Implementation</td>
<td>43</td>
</tr>
<tr>
<td>4.3 Algorithms</td>
<td>46</td>
</tr>
<tr>
<td>4.4 Performance Evaluation Metrics</td>
<td>48</td>
</tr>
<tr>
<td>5 Result and Discussion</td>
<td>51</td>
</tr>
<tr>
<td>6 Conclusions</td>
<td>57</td>
</tr>
</tbody>
</table>
Bibliography .............................................................. 59
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Load disaggregation by Hart(1992) [36]</td>
<td>2</td>
</tr>
<tr>
<td>2.1</td>
<td>Meta analyse about the impact of various feedback methods over behaviour change [28]</td>
<td>7</td>
</tr>
<tr>
<td>2.2</td>
<td>Signature Taxanomy for appliance identification by [36]</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Power consumption pattern of the Philips Ceiling Light</td>
<td>14</td>
</tr>
<tr>
<td>2.4</td>
<td>Power consumption pattern of the Washing Machine</td>
<td>15</td>
</tr>
<tr>
<td>2.5</td>
<td>Power consumption pattern of the iPad Charger</td>
<td>15</td>
</tr>
<tr>
<td>2.6</td>
<td>Power consumption pattern of the Refrigerator</td>
<td>16</td>
</tr>
<tr>
<td>2.7</td>
<td>Difference between AI, ML and DL</td>
<td>23</td>
</tr>
<tr>
<td>2.8</td>
<td>Architecture of Artificial Neural Network</td>
<td>24</td>
</tr>
<tr>
<td>2.9</td>
<td>Sigmoid graph</td>
<td>25</td>
</tr>
<tr>
<td>2.10</td>
<td>Tanh graph</td>
<td>26</td>
</tr>
<tr>
<td>2.11</td>
<td>RELU graph</td>
<td>26</td>
</tr>
<tr>
<td>3.1</td>
<td>Flow chart of Proposed approach</td>
<td>31</td>
</tr>
<tr>
<td>3.2</td>
<td>Architecture of 1D-CNN network in our model</td>
<td>32</td>
</tr>
<tr>
<td>3.3</td>
<td>Pictorial representation of proposed method</td>
<td>34</td>
</tr>
<tr>
<td>3.4</td>
<td>Appliance wise energy consumption distribution of house 1 in REDD</td>
<td>35</td>
</tr>
<tr>
<td>3.5</td>
<td>Power consumption pattern of Dishwasher of house 1 in REDD</td>
<td>35</td>
</tr>
<tr>
<td>3.6</td>
<td>Power consumption pattern of Refrigerator of house 1 in REDD</td>
<td>36</td>
</tr>
<tr>
<td>3.7</td>
<td>Power consumption pattern of Washing Machine of house 1 in REDD</td>
<td>36</td>
</tr>
<tr>
<td>3.8</td>
<td>Power consumption pattern of Microwave of house 1 in REDD</td>
<td>36</td>
</tr>
<tr>
<td>4.1</td>
<td>Energy consumption distribution by the appliances in house 1</td>
<td>39</td>
</tr>
<tr>
<td>4.2</td>
<td>Frequency of power reading of each appliances and main meter</td>
<td>40</td>
</tr>
<tr>
<td>4.3</td>
<td>Good section of refrigerator in house 1</td>
<td>40</td>
</tr>
<tr>
<td>4.4</td>
<td>Energy consumption distribution by the appliances in house 2 UK-dale</td>
<td>41</td>
</tr>
<tr>
<td>4.5</td>
<td>Frequency of power reading of each appliances and main meter UK-dale</td>
<td>42</td>
</tr>
<tr>
<td>4.6</td>
<td>Good section of refrigerator in house 2 UK-dale</td>
<td>42</td>
</tr>
<tr>
<td>4.7</td>
<td>Architecture of NILMTK</td>
<td>44</td>
</tr>
<tr>
<td>5.1</td>
<td>Disaggregation performance of Proposed method</td>
<td>56</td>
</tr>
</tbody>
</table>
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Publicly available NILM datasets</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>List of different types of activation functions</td>
<td>27</td>
</tr>
<tr>
<td>3.1</td>
<td>Number of activation of appliances per home in REDD.</td>
<td>37</td>
</tr>
<tr>
<td>3.2</td>
<td>Selected Houses for Training and Testing.</td>
<td>37</td>
</tr>
<tr>
<td>4.1</td>
<td>Sub-meters installed in each house in REDD dataset.</td>
<td>39</td>
</tr>
<tr>
<td>4.2</td>
<td>Sub-meters installed in each house in UK-dale dataset.</td>
<td>41</td>
</tr>
<tr>
<td>4.3</td>
<td>Confusion Matrix</td>
<td>50</td>
</tr>
<tr>
<td>5.1</td>
<td>Disaggregation performance on houses not seen during training.</td>
<td>52</td>
</tr>
<tr>
<td>5.2</td>
<td>Performance of appliance operating state identification on houses not seen</td>
<td>52</td>
</tr>
<tr>
<td>5.3</td>
<td>Identification performance on houses seen during training in REDD.</td>
<td>53</td>
</tr>
<tr>
<td>5.4</td>
<td>Disaggregation performance on houses seen during a training in UK-dale.</td>
<td>53</td>
</tr>
<tr>
<td>5.5</td>
<td>Performance of appliance operating state identification on houses seen during training on RESIDE dataset</td>
<td>54</td>
</tr>
<tr>
<td>5.6</td>
<td>Performance of appliance operating state identification on houses unseen</td>
<td>54</td>
</tr>
<tr>
<td>5.7</td>
<td>A total number of the trainable parameter of each model.</td>
<td>55</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This chapter will first discuss the background of non-intrusive load monitoring and how it can be helpful, problems faced by current state-of-the-art NILM methods, the main contribution of this study, and at last outline of this thesis.

1.1 Background

Currently, residential energy consumption accounts for more than 20% of the total energy consumption worldwide, of which two-third is from developing countries [42]. Residential energy consumption is rapidly growing, with an increase of 33.7% in the past two decades. With economic development, urbanization, and improved people’s living standards in developing countries, residential energy demand will increase further [63]. The International Electrotechnical Commission (IEC) has stated that electricity’s intelligent and economic use will be the most crucial factor in solving energy problems as the primary energy source. Thus, efficient and sustainable utilization of energy has been an essential area of research in recent times. Appliance load monitoring helps to reduce energy wastage by creating awareness among users. The detailed consumption pattern also enables utility companies in effective load management. Literature shows that energy feedback information provided by smart meters can enable consumers to reduce consumption between 5% to 15% [24].

Individual appliance energy consumption is monitored by two approaches, intrusive load monitoring and non-intrusive load monitoring. Intrusive load monitoring uses smart plugs and smart strips on each device to monitor their energy consumption. Non-intrusive load monitoring (NILM) or energy disaggregation is the computational technique to estimate the individual
appliance power consumption from the main meter aggregated data, as shown in Figure 1.1. NILM is a cost-effective approach for giving appliance-wise energy consumption feedback. The simple use case of NILM is producing an individual utility bill from the single, smart main meter. NILM is cost-effective as compared to device-level monitoring.

![Figure 1.1 Load disaggregation by Hart(1992) [36]](image)

Figure 1.1 Load disaggregation by Hart(1992) [36]

Aggregated data or main meter data consists of a large number of appliances, anonymous power sources, appliances with the same power consumption, and the switching ON/OFF several devices at the same moment. Therefore, NILM or energy disaggregation is a single-channel blind signal separation (BSS) problem. BSS plays a role when it is necessary to isolate the speech of one person in a cocktail party (cocktail party paradigm) [54], to separate musical signals or even medical imaging and image processing. The energy disaggregation problem at any time $t$ is shown as:

$$X_t = \sum_{i=1}^{n} y_i^t + \sigma(t)$$

(1.1)

Where $\sigma(t) = $ anonymous power source or noise. A given sequence of main power consumption $X = \{X1, X2, ..., X_T\}$ corresponding to $N$ active appliances at time $t = \{1, 2, 3, ..., T\}$. Thus, the objective is to infer the power contribution $y_i^t$ from the device $i \in \{1, 2, 3, ..., N\}$ at time $t$. 

* Hart (1992)
The NILM can perform on the two types of aggregated data high frequency data (2 Hz to 50 KHz) and low frequency data (1 sec to 1 day). Numerous variations in features and algorithms have led to the development of various NILM techniques. Recently deep learning has found wide acceptance in the area of energy disaggregation. Deep learning techniques automatically learn feature representations from the data. So, we don’t need any additional handcrafted features to perform energy disaggregation. The major challenge for NILM is building generalizable NILM models that perform accurately in real-time disaggregation across similar homes. NILM is taking more and more space in current research due to the development of smart grids and the massive deployment of smart meters. In the literature author’s use many different names to refer to ‘energy disaggregation’ such as:

- NILM - Non-intrusive load monitoring [5]
- NIALM - Non-intrusive appliance load monitoring [3]
- NIALMS - Non-intrusive appliance load monitoring systems [10]
- ALM - Appliance load monitoring [87]

The main focus of NILM is to make real-time appliance level load monitoring cost-effective by eliminating smart plugs and replacing them with a single, smart main meter. The NILM helps users reduce energy consumption by giving individual appliance feedback, help operators to manage grids, and identifying faulty appliances, and survey appliance usage behavior. Nowadays, commercial companies like Bidgely, Smappee, Powerly, and many others offer NILM products worldwide.

1.2 Motivation

The primary motivation of this thesis is to develop a robust NILM algorithm with which users can view real-time disaggregated data. The real-time disaggregated data allows the user to determine which appliance consumes most of the electricity and shares of each appliance in the total electricity consumption. This kind of information leads the user to reduce their energy...
consumption by 5% to 15% [24]. The appliance level feedback can also help users identify the faulty appliances and the appliances turned on unnecessarily. We will discuss the benefits of NILM in detail in the literature review section.

1.3 Problem Statement

While exploring the domain, we found three problems that low-frequency NILM algorithms face. The first problem is the low identification and power retrieval accuracy of multi-state appliances such as washing machines, microwaves, refrigerators, and dishwashers. The second problem is poor transfer learning of the NILM algorithm among similar homes. A third problem is a high number of model trainable parameters, leading to high computational complexity.

1.4 Contribution of this thesis

The main objective of this thesis is to provide a novel NILM algorithm on low sampled aggregated data (1 min sampled) using deep learning techniques for low power IoT (Internet of Things) devices.

The main contribution of this thesis are as follow:

1. Proposes NILM algorithm for retrieving power consumption of four different multi-state appliances such as dishwashers, Microwave, Refrigerator and Washing machines using low sampled aggregated data. We have used publicly available energy datasets to perform our experiment.

2. Conducted experiments to evaluate the generalizability of proposed algorithms. We have trained the algorithm on some homes and evaluated the performance by applying it to unseen homes.

3. Demonstrate the NILM architecture with lower computational complexity and less trainable parameters. We compared the number of trainable parameters with existing state-of-the-art NILM algorithms.
1.5 Outline of the thesis

This thesis is organised into six chapters. Following are the description of the chapters:

**Chapter 2** Provide a literature survey of the low sampled NILM algorithms. It provides why NILM is necessary, how to perform NILM, current state-of-the-art methods and research gaps.

**Chapter 3** present a proposed methodology and experiment setup of the research.

**Chapter 4** present the experimental framework which describes datasets, data-preprocessing, benchmarking algorithms and performance evaluation matrices.

**Chapter 5** present the evaluation of different NILM algorithms and proposed approach.

**Chapter 6** consists of a summary, conclusion of this present work and possible future work suggestions.
Chapter 2

Literature review on NILM

In this chapter, the first section will discuss the need for appliance-level energy feedback and how it can be helpful to users to reduce their energy consumption. The following section will discuss what NILM is, how NILM is performed, and the different types of methodologies and algorithms of NILM are discussed in detail. After discussing the current state-of-the-art methodologies, the research gaps of NILM algorithms are discussed. The last section will give a brief overview of deep learning.

2.1 Appliance Level Feedback

Currently, residential energy consumption (i.e., direct energy consumption at home) accounts for more than 20% of the total energy consumption worldwide, of which two-thirds is from non-Organisation for Economic Co-operation and Development (OECD) countries [42]. Residential energy consumption is rapidly growing, with an increase of 33.7% observed in the past two decades. With economic development, urbanization, and improved people’s living standards in non-OECD countries, residential energy demand is anticipated to increase further [63]. The International Electrotechnical Commission (IEC) has stated that the intelligent and economical use of electricity, as the primary energy source, will be the most crucial factor in solving the energy problem [1]. Thus, efficient and sustainable utilization of energy has been an essential area of research in recent times. For this reason, home energy management systems (HEMS) are becoming increasingly important to reduce energy consumption in households while maintaining occupants comfort.
HEMS offers advantages to both residential occupants as well as electricity suppliers. For residential occupants, HEMS can perform real-time monitoring of appliances, schedule various household appliances as per the user’s preference, and manage home renewable systems and energy storage. For electrical suppliers, the two-way communication enabled by the smart grid allows better management of the whole electricity network and implementation of demand response. Demand response (DR) is a change in electricity consumption of an electric utility customer to better match the demand for power with the supply. The reviews on HEMS can be found in [14, 69, 85].

![Average Household Electricity Savings (4-12%) by Feedback Type](image)

**Figure 2.1** Meta analyse about the impact of various feedback methods over behaviour change [28]

The first step of any HEMS is to monitor the electricity consumption of various appliances and give energy feedback to the users. Energy feedback offers an opportunity for users to reshape their energy usage and manage their energy consumption. It can reduce electricity
consumption by 5-15% as per [24, 62]. A meta-analysis over 15 years [28] shows that we can influence the decision of the people to decrease their consumption between 4% to 12% and only with the combination of real-time information and feedback at a daily/weekly rate, as described in Figure 2.1. The other benefits of appliance level feedback are briefly discussed below.

**The benefits of appliance level feedback are as follow:**

- **Generating individual appliance electricity bills to help users to save electricity.**
  It allows the user to find out which appliance consumes most of the electricity and shares of each appliance in the total electricity consumption. Appliance level feedback is one of the most efficient ways for users to reduce electricity bills. Literature shows that energy feedback information provided by smart meters can enable consumers to reduce consumption between 5% to 15% [24].

- **Real-time log of each appliance.**
  The real-time log helps the user to monitor the status of each appliance in real-time and what amount of energy it is consuming. The best use-case can be a mobile application that detects when you are about to leave a home and then checks to make sure the unwanted appliances like clothes iron are turned off before leaving. Bidgely offers a service that can tell which appliances are turned on in real-time.

- **Energy-saving recommendations.**
  Dis-aggregated data enables more accurate, and personalised recommendations to save electricity from the recommender systems.

- **Detection of faulty appliances.**
  Accurate dis-aggregated data is useful to check the status of appliances and the detection of faulty appliances. For example, if the identification of fridge deforestation cycle is more frequent than normal, it might suggest that the fridge seal is damaged and it should be repaired/replaced. The literature [19] shows how the ageing of the appliances can be detected using NILM.
• Demand response.
  The dis-aggregated load data can help to identify which loads are flexible enough for the demand-side response [71].

• Anomalous load detection.
  Anonymous load detection helps the user’s to report possible electricity threats in public or private buildings.

• Occupancy monitoring.
  From dis-aggregated data, it may be possible to infer the presence or absence of any occupant in a household by its power consumptions. It could detect illegal occupants when the user is away from the home or on vacation. This technology also has security and privacy implications.

• Helps the grid operator to predict accurate energy demand.

• Helps utility companies to segment their user’s wisely.

Monitoring of individual appliances in real-time can be done in two ways: intrusive load monitoring and non-intrusive load monitoring. Intrusive load monitoring/plug load monitoring can be done using smart plugs, smart meters and smart strips installed on individual appliances. Non-intrusive load monitoring can be done by using a single smart meter installed at the main meter. Based on data derived from the main meter, the NILM algorithm estimates the power consumption of each appliance. NILM approach is better than intrusive load monitoring, mainly because sub-metering installation is often expensive, difficult to upgrade, and involves certain privacy issues [39]. Next section will briefly describe what is NILM and how it is performed.

2.2 Non-Intrusive Load Monitoring (NILM)

The research on NILM was started by George Hart between the mid-1980s and mid-1990s at MIT [36]. George Hart is also referred to as the ‘godfather of disaggregation’. In his work, he gave a general idea of NILM and described how to extract the feature of each appliance for
disaggregation. He describes a ‘signature taxonomy’ which can be useful for appliance identification as shown in Figure 2.2. After extracting the feature, the NILM or energy disaggregation problem can be solved using two approaches: optimisation (non-event based approach) and pattern recognition (event-based approach).

![Figure 2.2 Signature Taxonomy for appliance identification by [36]](image)

George Hart points out that the optimisation approach is an NP-complete ‘weighted set’ problem and a precise solution is only achievable by enumerating every possible state. This is computationally impractical because if we have \( n \) appliances, each appliance can occupy any one of \( s \) states, we have \( s^n \) combinations and time complexity will increase exponentially as \( O(s^n) \). This section will further discuss how NILM is performed and different type of algorithms in details.

The NILM approach can be divided into mainly 3 stages:
1. Data Collection

2. Feature Extraction

3. Appliance Identification and Power Retrieval

2.2.1 Data Collection

This is the first step and one of the most important steps in NILM algorithms. It collects the aggregated data from the main meter using a smart meter. This aggregated data contains various features such as real power, apparent power, power factor, current signals, voltage signals, voltage-current (V-I) trajectories and many more [9]. The features are collected as per the requirement of the algorithm. The sampling rate of data collection can be divided into high frequency and low frequency. According to [36], a sampling rate of more than 1 Hz can be considered as the high-frequency sampling rate, and it can go up to 50 kHz. A sampling rate less than or equal to 1 Hz can be considered as a low sampling rate.

High-frequency Data

High-frequency data can be in the range of 2 Hz to 50 kHz. It allows us to obtain more detail about each appliance’s waveform, from the higher harmonics or the raw current and voltage waveform shape. It also enables us to capture electrical noise. Two-dimensional V-I trajectories are a viable method to identify load signatures in terms of features [37]. In [35], the author showed how high-frequency electromagnetic inference signals help to distinguish similar power appliances. In [50, 51], the author showed how the current waveform helps to identify and retrieve the power of each appliance in real-time. The works done on the high-frequency data by using different kinds of features and approaches are as shown in [9, 2, 75, 6, 59, 16]. The performance of NILM algorithms on the high-frequency data is better than on the low-frequency data due to the high resolution of features. However, high-frequency data need sophisticated hardware to capture the information at a high rate, leading to high hardware costs. We need more storage space to store high-frequency data, which leads to high storage space and increases the number of trainable parameters, making a model more...
computationally complex. Thus, NILM algorithms that use high-frequency data face the following three problems: High hardware cost, a requirement of large storage space, and a large number of trainable parameters. So to eliminate these problems, we are presenting a novel NILM algorithm using low-sampled data in this thesis.

**Low-frequency Data**

Low-frequency data can be in the range of 1 sec to 1 day. It captures the instantaneous value from the time series of power variables i.e. voltage \(v\) [45], current \(i\) [66], root mean square voltage \(v_{rms}\) [27], root mean square current \(i_{rms}\) [58], active power \(P\) [31], apparent power \(S\) [30], reactive power \(Q\) [21], power factor \(PF\) [70] at the given frequency. Apparent power \(S\) is considered as a likely feature to identify load signatures on low sampled data as shown in [53]. In [21, 45], reactive power and active power is used to identify the load signatures and retrieve the power consumption of each appliance. In [77], the author has used \(P\) and \(v_{rms}\) measurements to identify the load signatures on varying supply voltages. Most of the low sampled NILM algorithms are using 1-sec or 6-sec sampling rates for the disaggregation. To further reduce the time complexity and number of parameters, we are presenting a novel NILM algorithm on a 1 min sampled power data.

To design, test and benchmark high-performance energy disaggregation algorithms, NILM researchers need the availability of open-access energy consumption datasets. These datasets record the aggregated demand of the whole house as well as the ground truth of individual appliances and offer a real and noisy environment which can lead to more accurate algorithm design. The comparison of the various publicly available dataset with their characteristics is shown in Table 2.1.

**2.2.2 Feature Extraction:**

To disaggregate the power of each appliance from the aggregated data, the feature and power consumption pattern of each appliance needs to be understood. George Harts in his work [36], defined four types of appliances according to the their features as follows.
### Table 2.1 Publicly available NILM datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Location</th>
<th>Duration</th>
<th>No. of house</th>
<th>Devices</th>
<th>Resolutions</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>REDD [48]</td>
<td>USA</td>
<td>3-19 days</td>
<td>6</td>
<td>24</td>
<td>15 kHz</td>
<td>V and P</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(aggregated),</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.5 Hz and</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Hz (sub-meter)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UK-dale [46]</td>
<td>UK</td>
<td>499 days</td>
<td>5</td>
<td>5-54</td>
<td>16 kHz</td>
<td>P</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(aggregated),</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1/6 Hz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(sub-meter)</td>
<td></td>
</tr>
<tr>
<td>AMPDs [57]</td>
<td>Canada</td>
<td>365 days</td>
<td>1</td>
<td>19</td>
<td>1 min</td>
<td>V, I, F, P, Q, S and PF</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iAWE [12]</td>
<td>India</td>
<td>73 days</td>
<td>10</td>
<td>33</td>
<td>1 Hz</td>
<td>V, I, F, P and phase angle</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(aggregated),</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 Hz and</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1/6 Hz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(sub-meter)</td>
<td></td>
</tr>
<tr>
<td>BLUED [7]</td>
<td>USA</td>
<td>8 days</td>
<td>1</td>
<td>1</td>
<td>12 kHz</td>
<td>I, V, and state transition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(aggregated)</td>
<td></td>
</tr>
<tr>
<td>DRED [73]</td>
<td>Netherlands</td>
<td>6 months</td>
<td>3</td>
<td>21-26</td>
<td>1 Hz</td>
<td>P</td>
</tr>
<tr>
<td>BERDS [55]</td>
<td>USA</td>
<td>365 days</td>
<td>1</td>
<td>4</td>
<td>20 sec</td>
<td>P, Q and S</td>
</tr>
<tr>
<td>REFIT [61]</td>
<td>UK</td>
<td>2 years</td>
<td>20</td>
<td>11</td>
<td>8 sec</td>
<td>P</td>
</tr>
<tr>
<td>GREEND [60]</td>
<td>Austria/Italy</td>
<td>1 year</td>
<td>9</td>
<td>9</td>
<td>1 Hz</td>
<td>P</td>
</tr>
<tr>
<td>ECO [15]</td>
<td>Switzerland</td>
<td>8 months</td>
<td>6</td>
<td></td>
<td>1 Hz</td>
<td>Q</td>
</tr>
</tbody>
</table>

- Steady-state Devices (On/Off)
- Finite State Machine (FSM) or Multi-state Appliances,
- Infinite State or Continuously Variable (multi-state) Appliances
- Always On Appliances

**Steady-state Devices (On/Off)**

Steady-state devices or On/off type devices have binary states of power such as a toaster, light
bulb, fan and many more as shown in Figure 2.3, the power consumption pattern of the Philips ceiling light. When it is turned on, the power level is increased by 40 to 50 watts and remains constant till the light is turned off. Steady-state devices can easily be classified because of clear features but if there is two steady-state appliances with similar power consumption then it can be difficult to classify them separately.

![Figure 2.3](image.png)

**Figure 2.3** Power consumption pattern of the Philips Ceiling Light.

**Finite State Machine or Multi-state Appliances**

The second type are finite state machines (FSM) or multi-state appliances which have multiple states such as dimmable lamps, washing machine, microwave, and many more. As shown in Figure 2.4, the washing machine has three different types of cycles: heating, rinsing and drying. When the washing machine is turned on, its heating and rinsing cycle starts and consumes around 1000 Watts and after completing its first cycle it goes to the rinsing state and consumes around 200 Watts. As shown in figure 2.4, we can see that all the cycle power consumption is continuously fluctuating at different power thresholds. Thus, to classify the multi-state appliances, we need to observe the power consumption pattern and understand the features for a long period of time (at least 1-full cycle time). In [47], the author has stated that accuracy for classifying multi-state appliances is very low as compared to FSM and on/off type devices. So, for testing the proposed algorithm we are selecting only multi-state appliances in this thesis.

**Infinite State or Continuously Variable Appliances**

Infinite state or continuously variable appliances have an infinite or continuous number of
Figure 2.4 Power consumption pattern of the Washing Machine.

Figure 2.5 Power consumption pattern of the iPad Charger.

Always On Appliances

Always on multi-state appliances are always in operating states except special cases such as power supply outages. Appliances like a refrigerator are considered as always on appliances. As shown in Figure 2.6, a refrigerator has a periodic power consumption pattern with two dif-
ferent cycles; freezing and defrosting.

Based on these feature selection methods, the NILM algorithms can be divided into two categories; such as event-based algorithms and non-event based algorithms. Event-based algorithms first detect the event from the aggregated data and then perform disaggregation. The different types of approaches for event detection are mentioned in [8]. After event-detection, the algorithm observes the behaviour of the aggregated load, identifies and disaggregates the power accordingly. Event-based approaches are Decision tree, Long short term memory (LSTM), Support Vector Regression (SVR), linear regression, naive based approaches, artificial neural network (ANN) and many more. Non-event based algorithms don’t need event-detection, they perform disaggregation by learning the usage pattern of appliances. Non-event based approaches are Hidden Markov model (HMM), Expectation maximisation (EM), matrix factorisation and many more.

2.2.3 Appliance Identification and Power Retrieval:

To disaggregate the appliance load from the aggregated data, we need to identify the operating state of each appliance at the given time and retrieve their corresponding power consumption. It can be done by using two methods: optimisation methods and pattern recognition
Optimisation methods: The optimisation problem can be described as follows: a smart main meter signal is a time-series\( Y = \{y_1, y_2, y_3, \ldots, y_k\} \) where \( k \) is total number of samples and \( y_t \) is the main meter reading at time \( t \). We described the operating states (on/off) of each appliances with boolean vector as \( A = \{a_1, a_2, a_3, \ldots, a_n\} \) for \( n \) appliances. The each stats of multi-state appliances are considered as a separate appliance. The power consumption of each appliances is described in vector \( P = \{p_1, p_2, p_3, \ldots, p_n\} \). So the total power consumption at \( y_t \) at time \( t \) is the sum of the power consumption of all active appliances.

\[
y_t = \sum_{i=1}^{n} a_i p_i + e
\]  

\( e \) is error term. If the power consumption of each appliance is known then disaggregation can be stated as a combinatorial optimisation problem where we try to find state vector \( A_t^* \) such that:

\[
A_t^* = \text{ArgMin} \left| y_t - \sum_{i=1}^{n} a_i p_i \right|
\]  

George Hart in his work [36] points out that the optimisation approach is an NP-complete ‘weighted set’ problem and a precise solution is only achievable by enumerating every possible state. This is computationally impractical because if we have \( n \) appliances, each appliance can occupy any one of \( s \) states, we have \( s^n \) combinations and time complexity will increase exponentially as \( O(s^n) \). Thus, the majority of NILM algorithms are based on the pattern recognition approach.

Pattern recognition methods: Due to advancement in the field of signal processing, computer vision and many other fields, a lot of pattern recognition approaches are developed using various machine learning and deep learning techniques. The NILM problem is similar to the cocktail party problem in which there are a number of people talking at the same time, and from that we identify a single person voice. This pattern recognition approach is classified into two categories: unsupervised learning and supervised learning. Since we are working on
the low-frequency data, we will discuss the different types of pattern recognition approaches of NILM for low sampled data. Following sections will explain unsupervised and supervised NILM algorithms in detail.

**Unsupervised Algorithms:** Unsupervised algorithms learns patterns from untagged data. Unsupervised algorithms use Hidden Markov Model (HMM), clustering, Expectation-Maximization (EM), binary factorisation, principal component analysis (PCA), and many other probabilistic models. Unsupervised NILM algorithms use clustering technique such as mean-shift clustering scheme as shown in [79], HMM and its variants in these papers [53, 49], Dynamic Time Warping (DTW) based approach [84, 83], expectation maximisation based approach [51], and non-negative matrix factorisation based approach [52]. The HMM and its variant are widely used for modelling appliance consumption behaviour. It defines the number of hidden states in which the model can represent the operating state (i.e. on, off or any intermediate state) of each appliance and then disaggregate the data based on the operating states and aggregated data. The HMM-based models usually face three major problems in NILM:

1. **High time complexity:** In the case of Factored Hidden Markov Model (FHMM), the time complexity is $O(Mn^{M+1}T)$ where $M$ is the number of appliances, $n$ is the number of hidden states and $T$ is the length of the observation. As the number of appliances increases, the time complexity increases exponentially. This results in low classification performance.

2. **Difficult to classify multi-state appliances:** Multi state appliances have multiple states of power consumption during their usage time. To classify them, we need to learn long-range patterns. The existing HMM and its variants are a first-order Markov process; they cannot accurately classify the multi-state appliances using the low sampled aggregated data.

3. **Difficult to classify appliances with similar power consumptions:** HMM and its variants are trained based on the markov chain. To classify the appliances with similar power consumption, HMM and its variants use the same markov chain. So, it is difficult to distinguish both appliances.
Supervised Algorithms: Supervised learning is learning a function that maps an input to an output based on the set of training examples. Supervised algorithms use clustering techniques, support vector machine, neural networks, decision trees, HMM, linear regression, logistic regression, naive Bayes approaches, and many other models. Some of the common supervised techniques applied to the low sampled NILM are an ANN or multi-layer perceptron [20, 18], convolution neural network [26, 45, 82], autoencoder [45], deep neural network [17, 47, 81], support vector machines [80], k-nearest neighbour (K-NN) [32], decision tree [44], naive Bayes classifiers [33], DTW based [80], and HMM variants [56, 3]. The main problem faced by all these models is to identify the multi-state appliances and poor generalisation property. The supervised model works accurately only on the homes on which they are trained, not on the unseen homes. Due to the advancement of deep learning in artificial intelligence there are many deep learning algorithms that are used for energy disaggregation. Main benefit of deep learning algorithms over normal machine learning algorithm is that they don’t need handcrafted features. So, the feature extraction and event detection part is skipped while using deep learning based NILM algorithm.

In [45], the author has applied the three different deep learning models for energy disaggregation namely Long short term memory (LSTM), denoising autoencoder, and a feed forward network which regresses the start time, end time and average power demand of each appliance activation. They have used a sequence-to-sequence approach for performing disaggregation and modeled a separate model for each appliance. They tested the model performance on the REDD and UK-Dale dataset on 6 sec and 1 sec energy data respectively. They also demonstrated the generalizability of the model by training on some homes and testing on the unseen homes. We will consider this paper’s experiment for benchmarking and compare our results with them. In [81], the author demonstrated that the sequence-to-point approach is better than the sequence-to-sequence approach by taking the same model and same set of experiments.

- Sequence-to-point approach is widely used in speech recognition. In a sequence-to-point approach, the input is the window of the aggregated power data and the output is the single point power consumption of the target appliance corresponding to the midpoint of the window.
In a sequence-to-sequence approach, the input is the window of the aggregated power data and the output is the window of the power consumption of the target appliance.

In [81], the author demonstrated that the sequence-to-point approach is better than the sequence-to-sequence approach because the sequence-to-point captures the features such as change points and edges of the target appliance precisely from the aggregated data window. They have also demonstrated that the sequence-to-point approach is the current state-of-the-art deep learning approach by comparing results with other deep learning models and traditional NILM models.

The author in [81] demonstrated the sequence-to-point approach for Convolution neural networks and tested their model on REDD dataset. The authors in [26] have proposed a CNN based technique to identify appliances. The authors in [17] describe a hybrid model for energy disaggregation through deep feature learning (DFL) on the REDD. They compared three different disaggregation methods namely the convolutional neural network, 1D CNN-RNN, and long short-term memory (LSTM) and showed that the proposed 1D CNN-RNN model was performing better than the others. In this thesis, all of these deep learning methods will be used to compare the results of our proposed algorithm. Next section will demonstrate the existing supervised NILM methods for 1 min sampled data in detail and what problems are faced by them.

**Existing supervised NILM methods for 1 min sampled data:**

In [23, 49], the author used the FHMM and its variants on the Almanac of minutely power dataset (Ampds), DTW distance-based approach on the Ampds and REDD dataset by these papers [76, 80] and support vector regression on the Ampds and REDD dataset by this papers [80]. The FHMM based model needs prior distribution of each appliance and has high time complexity. DTW distance-based models and SVR based models have better performance on the on/off appliances but have poor performance on the multi-state appliances. All these papers have used the Ampds dataset, and all these papers have trained their algorithm on 80% of data and tested on 20% of data of the same home. As the Ampds dataset has energy consumption data of only one home so, the papers which use Ampds dataset for evaluating their algorithms but don’t evaluate generalizability across different homes. Thus, this thesis proposed the NILM algorithm for 1 min sampled data and evaluate its performance on REDD
and UK-dale dataset.

**Recent developments in NILM algorithms.**

In [65], the author used an attention-based deep neural network to improve the generalization capability of the overall architecture by including an encoder-decoder with a tailored attention mechanism in the regression subnetwork. They tested the proposed method on REDD and UK-dale datasets. In [74], the author used a multilevel LSTM autoencoder which showed how to account for the temporal variability of input signals in a multi-label classification framework and tested on REDD dataset with current state-of-the-art deep learning algorithms. In [43], the author demonstrates the sequence to point learning based on the bidirectional dilated residual network to address exploding gradient and network degradation problems on multi-state appliances. They tested the proposed algorithm on a 1 sec sampled REDD and UK-dale dataset. In [11], the author presented the lightweight NILM using pruned sequence to point approach on the REFIT dataset and showed that sequence-to-point (seq2point) learning is one of the most promising methods for tackling NILM. In [29, 86, 64], the author’s have presented a different type of deep learning based approach on 1 sec sampled aggregated data and tested on REDD and UK-dale datasets. In recent times there have been lots of development in the NILM algorithm based on deep learning frameworks [40, 41, 78].
2.2.4 Research Gaps

Based on the detailed literature review of the NILM algorithms we have identified the four research gaps for low sampled NILM algorithms. The research gaps are as follow:

1. There are less number of non-intrusive load monitoring studies on the 1 min sampled aggregated data. The majority of studies done on 1 min sampled aggregated data were done on the Ampds dataset. The Ampds dataset contains data of only one home which makes it impossible for researchers to test the generalizability of their models across different houses.

2. The performance and generalizability of the NILM models on the 1 min sampled aggregated data is low compared to the 1 sec and 6 sec sampled data due to less number of features. We need to improve the performance of the NILM models on 1 minute sampled aggregated data to make models more accurate and practicable.


4. The NILM algorithms which use deep learning, ANN, Support vector machine, or HMMS have high computational complexity and need large storage space for performing disaggregation.

This thesis proposes the NILM algorithm for low-sampled aggregated data using deep learning techniques to address these research gaps. So, the following section will give a brief idea about what is deep learning and how it works and what are different types of components in it.
2.3 Deep Learning

This section will first discuss deep learning, ANN and its working. After that, it will describe the architecture of ANN different types of functions in ANN. At last, it will discuss convolution neural networks and recurrent neural networks in detail.

Artificial intelligence refers to the simulation of human intelligence in a machine programmed to think like humans and mimic their actions. An artificially intelligent machine rationalizes and takes action that has the best chance of achieving a particular goal. Machine learning is a subset of artificial intelligence in which a computer program automatically learns how to complete a task without being explicitly programmed. In traditional Machine learning techniques, most of the applied features need to be identified by a domain expert to reduce the complexity of the data and make patterns more visible to learning algorithms to work. Deep Learning algorithms try to learn high-level features from data in an incremental manner, eliminating the need for domain expertise and hardcore feature extraction.

![Figure 2.7 Difference between AI, ML and DL.](image)
2.3.1 Artificial Neural Network

An ANN is a directed graph where the nodes are artificial neurons and the edges allow information to pass from one neuron to another (or the same neuron in a future time step). Usually, neurons are arranged into layers such that each neuron of layer $m$ is connected to every neuron of layer $m + 1$ as shown in the Figure 2.8. The connection between two neurons is called weights. Each neuron takes the input from other neurons and applies the activation function (non-linearity) to the weighted sum of input and then passes the output. The leftmost layer of neurons is called the input layer and the rightmost layer of neurons is called the output layer and all the remaining layers between them are called the hidden layer.

![Architecture of Artificial Neural Network](image)

**Figure 2.8** Architecture of Artificial Neural Network.

The ANN or multilayer perceptron is called a feed-forward network because information flows from the input layer, through all hidden layer and to the output layer. The learning of the ANN is done by fine-tuning the weights of each neuron by using the backpropagation. Backpropagation computes the gradient of the loss function with respect to the weights of the network.
Functions in an ANN:

This part will explain the different type of functions in ANN. The function in ANN are as follow:

Transfer function It describes the relationship between the input and output of a system by using the polynomial equation. In ANN, each neuron has a simple transfer function that gives the weighted sum of the inputs. The output from the transfer function is further passed to the activation function as shown in Figure 2.8.

Activation function: An activation function is also known as the transfer function because it also maps the input node to the output node in a certain fashion. The main purpose of the activation function is to introduce non-linearity into the network that allows us to model the target variable which varies non-linearly with its feature or input variables. It is also continuously differentiable which allows gradient-based optimisation methods. The types of non-linear activation function and its behaviour are:

- **Logistic and Sigmoid** It maps any size of the input to output in a range \([0,1]\). It is used in neural networks for classification problems. The graph of Sigmoid function is shown in Figure 2.9. Sigmoid function equation is shown in equation 2.3

\[
f(x) = \frac{1}{1 + e^{-x}}
\]

(2.3)

Figure 2.9 Sigmoid graph.

- **Tanh** It maps any size of the input to output in a range \([-1,1]\). It is mainly used in the classification between two classes. The graph of Tanh function is shown in Figure 2.10.
Tanh function equation is shown in equation 2.4.

\[ f(x) = \left( \frac{2}{1 + e^{-2x}} \right) - 1 \]  

(2.4)

**Figure 2.10** Tanh graph.

- **RELU (Rectified linear unit)** It is a linear function that output the input directly if it is positive; otherwise, it will output zero. It has become the default activation function for many neural networks because a model that uses it is easier to train and often achieves better performance. The graph of RELU function is shown in Figure 2.11.RELU function equation is shown in equation 2.5.

\[ f(x) = \max(0, x) \]  

(2.5)

**Figure 2.11** RELU graph.
- **Softmax** It is used to impart probability when you have more than one type of output. It gives the probability for each class and the most probable output compared to other outputs is selected. It is mainly used for multi-class classification problems. Softmax function equation is shown in equation 2.6.

\[
f(x) = \frac{e^{x_j}}{\sum_{k=1}^{k} e^{x_k}}
\]  

(2.6)

**Loss function**

The loss function calculates the prediction error of any kind of neural network. It is used to calculate the gradients and the gradient are used to update the weights of the neural network. There are different types of loss function for different objectives as shown in Table 2.2.

<table>
<thead>
<tr>
<th>Name</th>
<th>Purpose</th>
<th>Equation</th>
<th>Use-case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary cross-entropy</td>
<td>Binary Classification</td>
<td>[ L(\Theta) = (-1/N) \sum_{i=1}^{N} (y_i)(\log(\hat{y}_i)) + (1-y)(\log(1-\hat{y}_i)) ]</td>
<td>The output value should be passed from the sigmoid activation function.</td>
</tr>
<tr>
<td>Categorical cross-entropy</td>
<td>Multi-class classification</td>
<td>[ L(\Theta) = (-1/N) \sum_{i=1}^{N} \sum_{c=1}^{C} (1_{y_i \in C_c})(\log(p_{model}[y_i \in C_c])) ]</td>
<td>The output value should be passed from the softmax activation function.</td>
</tr>
<tr>
<td>Mean square error</td>
<td>Regression</td>
<td>[ L(\Theta) = (1/N)(\sum_{i=1}^{N} (y_i - \hat{y}_i)^2) ]</td>
<td>The output value can be passed through any linear activation function.</td>
</tr>
<tr>
<td>Mean absolute error</td>
<td>Regression</td>
<td>[ L(\Theta) = (1/N)(\sum_{i=1}^{N}</td>
<td>y_i - \hat{y}_i</td>
</tr>
</tbody>
</table>

**Optimizer:**

Optimizers are an algorithm or method which changes the attributes of the neural network such as weights and learning rate in order to reduce losses and provide the most accurate solution. It is also referred as optimisation algorithms. The different type of optimisation algorithm
and its advantages are as follow:

**Gradient Descent:**

It is a first-order optimisation function that depends on the first order derivative of the loss function. During the backpropagation, the loss is transferred from one layer to another and the weights are modified based on the losses so that loss can be minimized. Equation of gradient descent is shown below.

$$\theta = \theta - \alpha \cdot \Delta J(\theta) \quad (2.7)$$

It is easy to implement, compute, and understand. The weights are changed after calculating the gradient of the whole dataset, which requires a lot of time and memory. Sometimes it converges in local minima. There are many variants of gradient descent to conquer such errors such as Stochastic gradient descent and batch gradient descent.

**Momentum:**

Momentum was invented for reducing the high variance in stochastic gradient descent and softening the convergence. It accelerates the convergence towards the relevant direction and reduces the fluctuation just by adding an extra hyper-parameter in the stochastic gradient descent formula.

There are many other variants of momentum and gradient descent that improves the performance of the optimizer such as Adams, Adagrad and many more.

2.3.2 **Convolution Neural Network**

Convolution neural networks (CNN) are commonly used for image processing. It consists of three layers namely, Convolution layer, Pooling layer, and Fully connected layer. The architecture of CNN is based on the alteration of convolution layers and pooling layers [4]. The convolution layer consists of neurons that are responsible for extracting different important features from the input data. After the convolution layer, the pooling layer comes which reduces the number of connections to the following layers. We are using a Max pooling layer. Max
pooling layer returns the maximum values obtained in the filters. In the end, the fully connected layer converts all the neurons of the interior layer to the output neurons, which in turn represent the classes to be classified [4]. The convolution layer is used for dimension reduction and extracting valuable features from any raw data. In computer vision, they use 2D images as inputs so they use 2D CNN. CNN captures small features better than the ANN or multi-layer perceptron so to capture the small important features of each appliance we have used the CNN. In this work, we are using 1D CNN because our input data is time-sequential data (1D data).

2.3.3 Recurrent Neural Network

From the previous paragraph, we see that the convolution neural network is a feed-forward network that maps from the input vector to a single output vector. In this neural network, when it is fed with new input, the memory of the previous input is removed. Recurrent neural networks (RNNs) allow cycles in the network graph such that the output from neuron \( i \) in layer \( l \) at time step \( t \) is fed via weighted connections to every neuron in layer \( l \) (including neuron \( i \)) at time step \( t + 1 \) [45]. This allows RNNs to map from the entire history of the inputs to an output vector and makes RNNs well suited to sequential data. RNNs can suffer from the “vanishing gradient” problem [38] where gradient information disappears or explodes as it backpropagation. The solution for this problem is “Long short term memory” (LSTM) architecture [38]. LSTM uses a “memory cell” with a gated input, gated output and gated feedback. The intuition behind LSTM is that it is a differentiable latch (where a latch is the fundamental unit of a digital computer’s RAM). LSTMs have been used with success on a wide variety of sequence tasks including automatic speech recognition [34] and machine translation [72].
Chapter 3

Proposed Methodology and Experiment

The main objective of this thesis is to provide a novel NILM algorithm on low sampled aggregated data (1 min sampled) using deep learning techniques. In this chapter we are presenting our proposed methodology and experiments to test the performance and generalizability of our proposed method. We are training and testing our model by using real aggregated data instead of adding any artificial one.

3.1 Proposed Methodology

This section will explain the proposed approach and the methodology in detail. The approach is the combination of the 1D-CNN (convolution neural network) and LSTM (Long short term memory). 1D-CNN is used for identifying the operation state of the appliance and LSTM is used for power retrieval of that appliance according to its consumption pattern. To capture all possible dependencies, we have trained 1D-CNN and LSTM models separately for all the devices. The proposed method is suitable for the low-frequency data and requires low computational power once the model is trained. Once the neural network is trained, it does not need ground truth data from each house to predict. For training our approach includes three steps, same as in [80]. First, we extract the power consumption pattern signature of each appliance and after that we estimate the power value of each appliances. The aggregation power is used to train the 1D-CNN network to identify the operating state of each appliance and at last, we used the active operating states for training LSTM for retrieving the power value. Figure 3.1 shows the flow chart of the proposed method. For testing we feed an aggregated power
window to the 1D-CNN trained model and retrieve appliance operating state of each appliance and at last, we feed the operating state from 1D-CNN to LSTM to get the power value of each appliance.

![Flow chart of Proposed approach.](image)

**Figure 3.1** Flow chart of Proposed approach.

### 3.1.1 Appliance Signature Extraction

Every appliance shows a unique power consumption pattern while operating. In this section, we will extract the power pattern signature of each appliance which will be used to predict power consumption. We extract the continuous sequence of power consumption of the appliances when their operation state is active. These extracted sequences are further used to train the LSTM model.
### 3.1.2 Appliance Operating State Identification

As discussed in the literature review, the sequence to point approach is better than the sequence to sequence approach. The sequence-to-point method trains a neural network to predict the output corresponding to the window’s midpoint, not the output of the entire window. Thus, the input of the network is the main window $X_{t:t+W-1}$, and the output is the midpoint element $y_{\tau}$ of the corresponding window of the target appliance, where $\tau = t + \lfloor W/2 \rfloor$. This method is widely used to model the distribution of speech and image [68]. It assumes that the mid-point element is represented as a non-linear regression of the mains window [81] because the output corresponding to the mid-point is highly dependent on the points before and after that mid-point. The paper [81] shows explicitly in their experiments that the change points (or edges) in the mains are the features that the network uses to infer the states of the appliance. We have tested the window length of 15, 20, 25, and 30 and find the F1-score of each. Window length of 20 gives the best result of 0.91 average F1-score on all the houses. Thus, we are using the sequence to point approach by providing the input of an array of length 20 to the network and predict the operating state corresponding to the 10th value of the array. In this paper, we are using the 1D-CNN model for identifying an operating state of each appliance i.e. every appliance has a separate 1D-CNN model. We are using categorical cross-entropy function as a loss function to get better results. Figure 3.2 shows the architecture of the 1D-CNN used in this thesis.

![Figure 3.2 Architecture of 1D-CNN network in our model.](image-url)
3.1.3 Appliance Power Retrieval

We are using a sequential LSTM for predicting the power consumptions because the power data is time-series data. The input of the LSTM would be the past five states of the appliances and output will be the power value for the present time step. We are using the Rectified Linear unit (RELU) as an activation function at each layer and mean square error as a loss function in our model. We are training each model for each appliance. Below is the architecture of LSTM used in this thesis.

**LSTM model:**

1. Input layer (length =5)

2. Sequential LSTM layer (N= 50, activation = RELU)

3. Sequential LSTM layer (N= 50, activation = RELU)

4. Dense Layer

5. Loss Function: Mean Square Error (MSE)

**Combined Schematic of Whole Model:**

The Figure 3.3 shows that the model takes the input of n (1*20) aggregated power readings and fed to the 1D-CNN. 1D-CNN gives the operating state at each timestep i.e. n (1*1). This operating state is indexed based on the active time i.e. if the device is inactive, index would be 0, if the device is active and previous 4 timestep is also active, index would be 5. After indexing the data-pre-processing for LSTM is done. We are giving past 5 indexes to LSTM to predict power value at the current timestamp. The list of dimension n(1*1) is changed to n(1*5) and fed to the LSTM which returns the active power. Thus, we fed aggregated power and got the power consumption by appliance. We are building appliance identification models for each appliance type so unseen appliances will not be identified.
3.2 Experiments

Our main motivation is to compare the model performance in terms of disaggregation and generalizability with state-of-the art algorithms and deep learning benchmarking NILM algorithms. To test the proposed model performance we divided the experiment into three stages and tested it on REDD, UK-dale and RESIDE dataset separately.

1. Appliance Selection

2. Training and Testing on Same House

3. Training on Some Houses and Testing on Unseen Houses.

Appliance Selection For giving the effective feedback we need to provide the appliance level energy consumption of appliances which accounts for the majority of the power consumption. As shown in Figure 3.4 we can see that Dish washer, refrigerator, washing machine and microwave consume the majority of power consumption. All these four appliances are multi-state appliances because all four appliances have more than one operating states as shown in Figure 3.5, 3.6, 3.7, and 3.8. Thus, we are considering these four appliances for testing the accuracy of our proposed methodology on multi-state appliances.
**Figure 3.4** Appliance wise energy consumption distribution of house 1 in REDD.

**Figure 3.5** Power consumption pattern of Dishwasher of house 1 in REDD.
Figure 3.6 Power consumption pattern of Refrigerator of house 1 in REDD.

Figure 3.7 Power consumption pattern of Washing Machine of house 1 in REDD.

Figure 3.8 Power consumption pattern of Microwave of house 1 in REDD.
Training and Testing on Same House: For testing the performance of the proposed method on multi-state appliances, we have taken the REDD and UK-dale dataset for training and testing purposes. There are 6 houses in the REDD dataset, 5 houses in the UK-dale dataset and 11 houses in the RESIDE dataset. We have divided each house data into 70:30 for training and testing. We have trained the separate model for each appliance and for each house and compared the result with Deyvisan [25], seq2seq autoencoder [45] and seq2seq LSTM [45].

Training on Some Houses and Testing on Unseen Houses: For testing the generalizability of the proposed algorithm we require an adequate number of appliance activation for training data. Thus, we are choosing the houses based on the adequate number of the training samples for training. Table 3.1 shows the activation of appliances per house. In the experiment, the appliances chosen to disaggregate are refrigerator, dishwasher, and microwave. These appliances are the most common household appliances in all six houses of REDD dataset, contributing the most towards the household’s power consumption of REDD dataset. Table 3.2 lists the houses that are used for training and testing. Mean square error (MSE) and mean absolute error (MAE) are used to evaluate the performance of disaggregation. The results of our proposed methods are compared with following papers algorithms seq2seq [45], seq2point [81], GLU-Res [22] and CNN algorithms [82].

<table>
<thead>
<tr>
<th>House No.</th>
<th>Refrigerator</th>
<th>Dishwasher</th>
<th>Microwave</th>
<th>Washing machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6561</td>
<td>1034</td>
<td>487</td>
<td>534</td>
</tr>
<tr>
<td>2</td>
<td>8739</td>
<td>247</td>
<td>134</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>8697</td>
<td>284</td>
<td>167</td>
<td>1036</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>219</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1399</td>
<td>54</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>8571</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Appliance</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refrigerator</td>
<td>2,3,5,6</td>
<td>1</td>
</tr>
<tr>
<td>Dishwasher</td>
<td>1,2</td>
<td>4</td>
</tr>
<tr>
<td>Microwave</td>
<td>1,2</td>
<td>3</td>
</tr>
</tbody>
</table>
Chapter 4

Experimental Setup

This chapter will discuss the dataset used for the evaluation of proposed method, data preprocessing techniques used for the experiments, current state-of-the-art benchmarking algorithms, and at last evaluation metrics for comparing the performance of the proposed methodology.

4.1 Datasets

After reviewing the publicly available datasets in table 2.1, we have chosen two datasets, one from America and one from Europe, for the experiment. The datasets are REDD and UK Dale. These datasets contain the aggregated energy consumption of the entire household and the ground truth energy consumption of individual appliances from multiple houses. Thus, it allows us to evaluate the disaggregation and transfer learning performance of the proposed algorithms. The datasets offer a real and noisy environment which allows us to evaluate the performance of the proposed algorithm. The description and visualization of both datasets are as follows.

Residential Energy Dis-aggregation Dataset (REDD)

The REDD dataset was released in 2011 [48], it is used on a large scale by the scientific community of the NILM field. It contains both aggregated and sub-metered power data from the six households for 3-19 days. The dataset is available in three frequencies: low_freq, high_freq, and high_freq raw. High_freq and high_freq raw contain voltage and current waveform recorded at 15 kHz for 2 houses. Low_freq contains power reading at 1Hz in each house
and appliance level reading at 3 second frequency. For our experiment we have downsampled low_freq dataset to 1 min. Number of sub-meters installed in each house is shown in the Table 4.1.

<table>
<thead>
<tr>
<th>House No.</th>
<th>Number of sub-meters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>26</td>
</tr>
<tr>
<td>6</td>
<td>17</td>
</tr>
</tbody>
</table>

The energy consumption distribution by the appliances in house 1 is shown in Figure 4.1:

![Figure 4.1 Energy consumption distribution by the appliances in house 1.](image)

Frequency of power reading of each appliances and main meter is shown in Figure 4.2. For performing NILM algorithm need to have continuous data, Figure 4.3 shows the good section of refrigerator.
Figure 4.2 Frequency of power reading of each appliances and main meter.

Figure 4.3 Good section of refrigerator in house 1.
UK-dale:

The first version of the UK-dale dataset was released in 2015 and the latest version was released in 2017 [46]. This dataset records the power demand from five houses. In each house they recorded both whole-house main power consumption every 1/6 seconds as well as power consumption of individual appliances every 6 seconds. They have also recorded the voltage and current waveform at 16kHz in 3 houses. It contains the main meter and sub-metered data of 5 years. Number of sub-meters installed in the each house is shown in Table 4.2.

<table>
<thead>
<tr>
<th>House No.</th>
<th>Number of sub-meter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>54</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>26</td>
</tr>
</tbody>
</table>

The energy consumption distribution by the appliances in house 2 is shown in Figure 4.4:

![Figure 4.4 Energy consumption distribution by the appliances in house 2 UK-dale.](image)
Figure 4.5 Frequency of power reading of each appliances and main meter UK-dale.

Figure 4.6 Good section of refrigerator in house 2 UK-dale.
RESIDE:

As a part of RESIDE project, we had collected the current data from 11 houses for 19 days in India. The dataset recorded the electricity consumption from each house using the Garud device every 1 minute. It contains the manually tagged air conditioner’s operating state data every minute. We have carried out two different sets of experiments on RESIDE dataset to demonstrate the performance of the proposed method. We have trained and tested on the same houses. We split training and testing data in a 70:30 ratio and trained separate models for each house. Furthermore, the second experiment was to train on some houses and test on unseen houses to verify the proposed method’s generalizability.

4.2 Data Pre-processing and Implementation

The Initial pre-processing of a dataset is done using Non-intrusive load monitoring Toolkit (NILMTK) [13]. The NILMTK is an open-source toolkit designed specifically for comparing different energy disaggregation algorithms on different publicly available datasets. The toolkit includes parsers for wide number of publicly available datasets, collection of pre-processing algorithms, set of statistics and visual function for describing datasets, two reference benchmarking disaggregation algorithms and suite of metrics for evaluation of algorithms [13]. The working and structure of NILMTK is shown in Figure 4.7.

Both dataset is down sampled from 3-sec data to 1 min data. All the gaps and NAN values in readings are detected and removed using NILMTK. After that, the longest continuous time sequence readings are identified with the help of a NILMTK functions. Appliance activations and their corresponding power consumptions are also extracted. As we are dealing with the continuous time- series data so it is necessary to obtain the continuous-time sequence. Thus, we extract all the continuous-time sequences, convert the length of the sequences into multiples of 20 by adding zeros or removing some values and store it separately. The training and testing data will be taken from these time sequences. We train 2 models for each appliance. Data pre-processing for each is explained below.

Data Pre-processing for 1D CNN: The 1D-CNN model is used for identifying the operating state of the appliances (0 or 1). 1D-CNN uses the sequence to point approach as men-
tioned in the section 3.1.2. It takes an input array of dimension \( n(1 \times 20) \) where \( n \) is the number of timestamps and gives output \( n(1 \times 1) \). For the endpoint of the full input sequence \( X = (x_1, x_2, ..., x_t) \) and \( y = (y_1, y_2, ......, y_t) \) we have added zeros at both the ends. The number of zeros to be added are calculated by \( \lceil Windowlength/2 \rceil \) so, 10 zeros at beginning and 10 zeros at end in our case. After adding the zeros, we slide a window of 20 from the starting point of each continuous sequence till the end and store the array in X and their corresponding output in Y.

**Data Pre-processing for LSTM:** The output from the 1D-CNN is the operating state of the appliances at the corresponding time step. After the pre-processing of the operating state we are feeding them to LSTM to predict the power value of that appliance at that time step. 1D-CNN gives the operating state at each timestep as \( n(1 \times 1) \). This operating state is indexed based on the active time, if the device is inactive, the index would be 0, if the device is active and the previous 4 timestep is also active, index would be 5. The power value and index is stored as shown below: Power reading of refrigerator: [148, 135, 129, 127, 127, 125] Power reading stored: [(1,148), (2,135), (3,129), (4,127), (5,127), (6,125)] To learn the power consumption pattern the LSTM takes the input array of past 5 appliance state indexes and gives power value as output. For training purpose, the length of input and output arrays is 5, the power at that
time respectively is shown below:

Input (X): \[\rightarrow\] Output (Y)

\[
\begin{align*}
[0, 0, 0, 0, 0] & \rightarrow 148 \\
[0, 0, 0, 0, 1] & \rightarrow 135 \\
[0, 0, 0, 1, 2] & \rightarrow 129 \\
[0, 0, 1, 2, 3] & \rightarrow 127 \\
[0, 1, 2, 3, 4] & \rightarrow 127 \\
[1, 2, 3, 4, 5] & \rightarrow 125
\end{align*}
\]

The window of length 5 is moved on operating index readings and fed to the LSTM to learn the power consumption pattern of each appliance. \(X = n(1*5)\) and \(y = n(1*1)\). Note that we are using only power readings when the appliance is active. For inactive power readings, we are taking the mean of the inactive power value of appliances from the dataset.

**Implementation:** The programming language that has been used for writing the code is python. The Keras library is used for 1D-CNN and LSTM. For data preprocessing we have used Pandas, Numpy and NILMTK. The models have been trained on the 12 hr free GPUs named Tesla K80 which are powered by Google Colaboratory using the TensorFlow backend. Google Colab is a research tool based on the Jupyter notebook environment for machine learning (ML) research and education. No setup is required to use a Jupyter notebook environment [67]. The inference is much cheaper when these DNNs are trained; it takes approximately a processing second per network of DNN for a week of aggregate data on a GPU. The neural nets learn efficiently if the input data have mean between 0 to 1. So we normalized our data using MinMax normalisation. MinMax normalisation is done using equation 4.1:

\[
x = (x - x_{min})/(x_{max} - x_{min})
\]  

(4.1)
4.3 Algorithms

This section will discuss the different state-of-the-art algorithms which are used to evaluate the performance of our proposed method. All the algorithm’s architecture are same as mentioned in the paper.

**Sequence to Sequence Convolution Neural Network [45]:**

This DNN learns a nonlinear regression map between a sequence of the mains power defined as a sliding window $Y_{t:t+W−1}$; and a sequence of appliance power defined as a window $X_{t:t+W−1}$ [45, 81]. Then, they model $X_{t:t+W−1} = F_s(Y_{t:t+W−1}) + \epsilon$, with $\epsilon$ is the dimensional Gaussian random noise $W$ that corresponds to $\sim N(0, \sigma^2_t I)$, and $F_s$ the neural network.

The exact architecture is as follows:

1. Input sequence with length $Y_{t:t+W−1}$
2. 1D Convolution: filters: 30; filter size: 10, activation function=ReLU
3. 1D Convolution: filters: 30; filter size: 8, activation function=ReLU
4. 1D Convolution: filters: 40; filter size: 6, activation function=ReLU
5. 1D Convolution: filters: 50; filter size: 5, activation function=ReLU
6. 1D Convolution: filters: 50; filter size: 5, activation function=ReLU
7. Fully connected: units: 1024, activation function=ReLU
8. Output: Number of units: $W$ similar to the input length

**Sequence to Point Convolution Neural Network [81]:**

Unlike the Seq2Seq, the Seq2Point only predicts the midpoint of an appliance given a window of the mains (the input). The mains readings are still defined as $X_{t:t+W−1}$ but the output becomes the midpoint element $y_\tau$ of the appliance’s power window $Y_{t:t+W−1}$, with $\tau = t + \lfloor W/2 \rfloor$. This DNN is based on the assumption that $y_\tau$ is represented as a non-linear regression of the mains window. [81]. It is expecting that the state of the midpoint element of that appliance to relate to the information of mains before and after that midpoint. Its main advantage is to make a single prediction for every midpoint element, instead of average of predictions for each
window. The demonstrations of [81] show that both Seq2Seq and Seq2Point are essentially posterior density estimators. It can be interpreted that both methods are minimizing a Monte Carlo approximation to KL-divergence.

The exact architecture is as follows:

1. Input sequence with length $X_{t:t+W-1}$
2. 1D Convolution: filters: 30; filter size: 10, activation function=ReLU
3. 1D Convolution: filters: 30; filter size: 8, activation function=ReLU
4. 1D Convolution: filters: 40; filter size: 6, activation function=ReLU
5. 1D Convolution: filters: 50; filter size: 5, activation function=ReLU
6. 1D Convolution: filters: 50; filter size: 5, activation function=ReLU
7. Fully connected: units: 1024, activation function=ReLU
8. Output: Output: Number of units:1, activation: linear

**Sequence to Sequence Gated Linear Unit Convolution Neural Network [22]:**

It is similar to sequence to sequence convolution neural network but instead of convolution block this network uses GLU convolution block to introduce more non-linearity in the model [22]. This network consist of GLU convolution block, max pooling block, fully connected layer, and residual block. The exact architecture is as follows:

1. Input sequence with length $X_{t:t+W-1}$
2. 1D Convolution: filters: 100; filter size: 4, activation function=ReLU
3. 1D Convolution: filters: 100; filter size: 4, activation function=ReLU
4. 1D Convolution: filters: 100; filter size: 4, activation function=ReLU
5. Fully connected: units: 100, activation function=ReLU
6. Residual Block: units: 50, activation function=ReLU
7. Residual Block: units: 50, activation function=linear
8. Residual Block: units: 50, activation function=linear
9. Fully connected: units: 100, activation function=linear
10. Output: Output: Number of units:1, activation: linear

**Sequence to Sequence Long Short Term Memory [45]:**

A recurrent neural network is the regular feed-forward neural network with memory. It maps the output vector with the entire history of inputs. It allows cycles in the network graph such that output from neuron i in layer l at time t is fed via weighted connection to every neuron in layer l, including neuron i at time t+1. Thus, this makes RNNs well suited to sequential data. In this paper [45], author have used sequence to sequence LSTM model to dis-aggregate the power consumption of each appliances. The exact architecture of the model is shown below:

1. Input sequence with length $X_{t:t+W-1}$
2. 1D conv (filter size=4, stride=1, number of filters=16, activation function=linear, border mode=same)
3. Bidirectional LSTM (N=128, with peepholes)
4. Bidirectional LSTM (N=256, with peepholes)
5. Fully connected (N=128, activation function=TanH)
6. Fully connected (N=1, activation function=linear)

**4.4 Performance Evaluation Metrics**

Evaluation metrics measure the quality of the statistical or machine learning models. This section will briefly discuss the metrics used to evaluate the NILM algorithms performance.
The metrics for NILM are mainly divided into two categories:

1. Regression Metrics.
2. Classification Metrics.

**Regression Metrics**

It measure how well the NILM algorithm can estimate or infer the value of power consumed by every appliance. We cannot calculate the regression model’s accuracy, so we measure the error between the predicted value and the ground truth. Two types of error are commonly used for evaluating NILM algorithms:

1. Mean Absolute Error (MAE)
2. Mean Square Error (MSE)

**Mean Absolute Error (MAE):** MAE is the most used metric to compare the performance of proposed methods with other research studies. It is calculated as the mean or average of the absolute difference between the predicted and the expected power values of the dis-aggregation algorithm.

\[
MAE = \frac{1}{T} \left( \sum_{i=1}^{T} |\hat{x}_t - x_t| \right)
\]  

(4.2)

**Mean Square Error (MSE):** It is calculated as the mean or average of the squared differences between the predicted and the expected power values of the dis-aggregation algorithm.

\[
MSE = \frac{1}{T} \left( \sum_{i=1}^{T} |\hat{x}_t - x_t|^2 \right)
\]  

(4.3)

**Classification Metrics**

Classification metrics are used to measure how well the NILM algorithm can predict the operating state of every appliance. We are comparing the operational state of each appliance at each time state whether that appliance is on/off. It can be measured by using confusion matrix and based on that we can get accuracy, precision, recall, and F1-score of the machine learning model.

**Confusion Matrix:** Confusion matrix is an specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one. Each row of the matrix
represents the instances in an actual class while each column represents the instances in a predicted class, or vice versa. In our case we have 2 classes, the device is on or off. Confusion matrix for NILM problem is made as shown in table 4.3:

<table>
<thead>
<tr>
<th>Confusion Matrix</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>TP</td>
<td>TN</td>
</tr>
<tr>
<td>False</td>
<td>FP</td>
<td>FN</td>
</tr>
</tbody>
</table>

True = Predicted Appliance state is same as ground truth
False = Predicted appliance state is not same as ground truth
Positive = Appliance is active
Negative = Appliance is in-active

Confusion matrix is used to calculate the precision, recall, accuracy and F1-score of the machine learning models.

**Precision:** Precision is how close the measured values are to each other. Precision is a metric which quantifies the number of true positive predictions made out of all positive predictions made. It is measured by using equation 4.4:

\[
Precision = \frac{TP}{TP + FP}
\] (4.4)

**Recall:** Recall is a metric which quantifies the number of true positive predictions made out of all positive predictions that would have been made. It is measured by using equation 4.5:

\[
Precision = \frac{TP}{TP + FN}
\] (4.5)

**F1-Score:** It is the harmonic mean of precision and recall as shown in equation 4.6.

\[
F1 - Score = \frac{2 \times P \times R}{P + R}
\] (4.6)

**Accuracy:** Accuracy measures the total number of predictions that were correct. It is calculated by using equation 4.7:

\[
Accuracy = \frac{TP + TN}{TP + TN + FN + FP}
\] (4.7)
Chapter 5

Result and Discussion

This chapter will compare the performance of the proposed method with the current state-of-the-art algorithms on the different sets of experiments. As mentioned in chapter 3, this thesis will perform three types of experiments on REDD, UK-dale and RESIDE dataset:

1. Training on some homes and testing on the unseen home.
2. Training and testing on the same house.
3. Compare the number of trainable parameters.

We will discuss the result of all these experiments as follow:

Training on some homes and testing on the unseen home.

This experiment address second research gap, "The performance and generalizability of the NILM models on the 1 min sampled aggregated data is low compared to the 1 sec or 6 sec sampled data due to fewer features. We need to improve the performance of the NILM models on 1 minute sampled aggregated data to make NILM more accurate and practicable." Experiment will evaluate the disaggregation performance of proposed method against the algorithms trained on 1 sec or 6 sec sampled data.

We have compared the MAE and MSE on unseen houses with seq2seq [45], seq2point [81], GLU-Res [22], and CNN [82]. All other algorithms are trained and tested on the one or 6-sec frequency data, and our algorithm is trained and tested on the 1 min (60 sec) frequency data. As shown in Table 5.1, our method outperforms the other four methods in power retrieval of microwave and dishwasher. The model reduces MAE and MSE by approximately 50% in microwave and dishwasher compared to the other four models. In the case of the refrigerator,
the proposed model performed better than seq2seq [45] and seq2point [81] but worse than GLU-Res [22] and CNN [82]. Table 5.1 demonstrates how well the proposed approach performs on the unseen data. Bold letters shows the best performance on the test data. Thus, we can see that the proposed method has a better capability for generalization.

<table>
<thead>
<tr>
<th>Model</th>
<th>Frequency</th>
<th>Metrics</th>
<th>Refrigerator</th>
<th>Microwave</th>
<th>Dishwasher</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seq2Seq</td>
<td>6 sec</td>
<td>MAE</td>
<td>30.6</td>
<td>33.3</td>
<td>19.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSE</td>
<td>2151.9</td>
<td>19292.8</td>
<td>14172.6</td>
</tr>
<tr>
<td>Seq2point</td>
<td>6 sec</td>
<td>MAE</td>
<td>28.1</td>
<td>28.2</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSE</td>
<td>2393.9</td>
<td>17483.5</td>
<td>15891.3</td>
</tr>
<tr>
<td>GLU-Res</td>
<td>1 sec</td>
<td>MAE</td>
<td>23.5</td>
<td>28.4</td>
<td>33.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSE</td>
<td>2197.4</td>
<td>25202</td>
<td>22301.1</td>
</tr>
<tr>
<td>CNN</td>
<td>1 sec</td>
<td>MAE</td>
<td>21.8</td>
<td>18.3</td>
<td>22.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSE</td>
<td><strong>1622.8</strong></td>
<td>17037.9</td>
<td>18658.5</td>
</tr>
<tr>
<td>Proposed_model</td>
<td>1 min</td>
<td>MAE</td>
<td>27.7</td>
<td><strong>12.3</strong></td>
<td><strong>13.9</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>MSE</td>
<td>4486.7</td>
<td><strong>9074.8</strong></td>
<td><strong>8144.5</strong></td>
</tr>
</tbody>
</table>

The other models have not given the performance based on appliance state identification for unseen houses. Table 5.2 shows the identification performance on unseen house.

<table>
<thead>
<tr>
<th>Appliance</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-Score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refrigerator</td>
<td>0.85</td>
<td>0.80</td>
<td>0.82</td>
<td>0.88</td>
</tr>
<tr>
<td>Microwave</td>
<td>0.61</td>
<td>0.80</td>
<td>0.69</td>
<td>0.99</td>
</tr>
<tr>
<td>Dishwasher</td>
<td>0.80</td>
<td>0.94</td>
<td>0.86</td>
<td>0.97</td>
</tr>
</tbody>
</table>

Thus, the proposed algorithm have better performance and generalizability on the 1 min sampled data as compared to state-of-the-art NILM algorithms on 1sec or 6-sec.

**Training and testing on the same house.**

This experiment address third research gap, "Lower identification accuracy on multi-state appliances and always-on appliances like washing machine, and refrigerator."

We have compared the operating state identification with Deyvisan [25], seq2seq autoencoder [45], and seq2seq LSTM [45] on the REDD dataset. All other algorithms are trained and tested on the one or 6-sec frequency data, and our proposed algorithm is trained and tested on the 1 min (60 sec) frequency. As shown in Table 5.3, the proposed method outperforms the other three methods. Bold letters shows the best performance on the test data.
We have compared the operating state identification with seq2seq autoencoder [45], seq2seq LSTM [45], and seq2seq rectangle [45] on the UK-dale dataset. All other algorithms are trained and tested on the 6-sec frequency data, and our proposed algorithm is trained and tested on the 1 min frequency. As shown in Table 5.4, the proposed approach outperforms the other three models. Bold letters shows the best performance on the test data.

Thus, the proposed algorithm have better performance on the multi-state appliances and always on appliances as compared to state-of-the-art NILM algorithms on REDD and UK-dale dataset.
We have demonstrate the performance of proposed method on RESIDE dataset by training and testing on same houses as shown in table 5.5.

**Table 5.5** Performance of appliance operating state identification on houses seen during training on RESIDE dataset

<table>
<thead>
<tr>
<th>House No.</th>
<th>Accuracy</th>
<th>F1-Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.99</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0.95</td>
<td>0.93</td>
<td>0.93</td>
<td>0.93</td>
</tr>
<tr>
<td>3</td>
<td>0.89</td>
<td>0.83</td>
<td>0.92</td>
<td>0.73</td>
</tr>
<tr>
<td>4</td>
<td>0.99</td>
<td>0.97</td>
<td>1</td>
<td>0.95</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>0.99</td>
<td>0.99</td>
<td>0.98</td>
<td>0.99</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>0.99</td>
<td>0.96</td>
<td>1</td>
<td>0.92</td>
</tr>
<tr>
<td>9</td>
<td>0.94</td>
<td>0.91</td>
<td>0.85</td>
<td>0.99</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

We have demonstrate the performance of proposed method on RESIDE dataset by training on some houses and testing on unseen houses as shown in table 5.6.

**Table 5.6** Performance of appliance operating state identification on houses unseen during training on RESIDE dataset

<table>
<thead>
<tr>
<th>Houses seen</th>
<th>Testing House</th>
<th>Accuracy</th>
<th>F1-Score</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>4,5,9</td>
<td>1</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
</tr>
<tr>
<td>4,5,9</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2,8</td>
<td>7</td>
<td>0.98</td>
<td>0.93</td>
<td>1</td>
<td>0.88</td>
</tr>
</tbody>
</table>

Thus, the proposed algorithm have better performance on the multi-state appliances with lower number of training data.

**Compare the number of trainable parameters.**

This experiment address forth research gap, ”The NILM algorithms which use deep learning, artificial neural network, Support vector machine, or HMMS have high computational complexity and need large storage space for performing disaggregation.”

We are comparing the number of trainable parameters of our model with seq2seq, seq2point, GLU-Res and CNN. All algorithms are trained on house 2 to 6 and tested on 1. As shown in table 5.5, the number of trainable parameters of our model is lesser than other models. Bold letters shows the best performance on the test data.
The results shown in the Table 5.1,5.3,5.4,5.5,5.6 and 5.7 shows that our approach is able to correctly detect the operating state of appliances with a higher accuracy, have good generalizability and lower size of model than any other state-of-art. Thus, all the research gaps are addressed in this section. Figure 5.1 represents the disaggregation result of proposed model on 500 continues test points of house 1. It has 2 parts, first part is aggregated reading, and the second part is the comparison with predicted and actual disaggregated power values. We have considered refrigerator, microwave, dishwasher and washing machine for disaggregation. Each diagram represent the different appliances.

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of trainable parameter (in Millions)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seq2seq</td>
<td>29.8</td>
</tr>
<tr>
<td>Seq2point</td>
<td>29.2</td>
</tr>
<tr>
<td>GLU-Res</td>
<td>1.2</td>
</tr>
<tr>
<td>CNN</td>
<td>0.738</td>
</tr>
<tr>
<td>Proposed method</td>
<td>0.070</td>
</tr>
</tbody>
</table>
Figure 5.1 Disaggregation performance of Proposed method.
Chapter 6

Conclusions

This thesis identifies the research gaps in NILM algorithms and proposes the load disaggregation algorithm for low sampled (1 min frequency) aggregated data using 1D-CNN and LSTM. The literature review section demonstrates the benefits of appliance-level feedback, working of non-intrusive load monitoring, discussed current state-of-the-art NILM algorithms, and identified the research gaps. The important gaps identified were low accuracy on multi-state appliances, poor generalizability of the model, and trainable parameters. Finally, we proposed the NILM algorithm by using 1D-CNN for appliance identification and LSTM for appliance power retrieval. We have carried out three different sets of experiments on the REDD and UK-dale datasets to compare the performance of proposed algorithms and current state-of-the-art algorithms. We chose six metrics to evaluate our model against the current state-of-the-art seq2seq, seq2point, GLU-res, and CNN. The results show that our proposed method can correctly detect the operating state with the 97% accuracy and 0.91 F1-score on the same houses. The proposed method gives 12.3 and 13.9 MAE on power retrieval of microwave and dishwasher on the unseen house. The results show that our proposed approach has good generalization ability, identification of operating state, and power retrieval of the multi-state appliance. The proposed method has 70K total parameters that consume 268 KB space to store the model weights. Thus, our approach has the advantage of quick execution in the real-time application, requires lower memory space, has high accuracy compared to state-of-the-art, and eligible for low cost embedded board such as ESP32 without any additional equipment for real-time feedback.
Related Publications

Bibliography


