NON INTRUSIVE REAL-TIME POWER MONITOR

Undergraduate final-evaluation project report submitted in partial fulfillment of the requirements for the
Degree of Bachelor of Science of Engineering
in

The Department of Electronic & Telecommunication Engineering
University of Moratuwa.

Main Supervisor:
Dr. Subodha Charles

Co Supervisor
Dr. Tharaka Samarasinghe

Group Members:
160422C-D.E.T.I. Nanayakkara
160464F-P.S.S.B. Pathiranage
160505J-R.M.I.P. Rajapakshe
160716G-P.M.N. Wijewardena

April, 2021

Approval of the Department of Electronic & Telecommunication Engineering

.....
Head, Department of Electronic
& Telecommunication Engineering

This is to certify that I/we have read this project and that in my/our opinion it is fully adequate, in scope and quality, as an Undergraduate Graduation Project.

Supervisor: Dr. Subodha Charles

Signature: .....

Date: .....

Co-supervisor: Dr. Tharaka Samarasinghe

Signature: .....

Date: .....

Declaration

This declaration is made on July 28, 2021.

Declaration by Project Group

We declare that the dissertation entitled "Non intrusive real time power monitor" and the work presented in it are our own. We confirm that:

- this work was done wholly or mainly in candidature for a B.Sc. Engineering degree at this university,
- where any part of this dissertation has previously been submitted for a degree or any other qualification at this university or any other institute, has been clearly stated,
- where we have consulted the published work of others, is always clearly attributed,
- where we have quoted from the work of others, the source is always given.
- with the exception of such quotations, this dissertation is entirely our own work,
- we have acknowledged all main sources of help.

2021/07/28
Date

.....
D.E.T.I. Nanayakkara (160422C)

.....
P.S.S.B. Pathiranage (160464F)

.....
R.M.I.P. Rajapakshe (160505J)

.....
P.M.N. Wijewardena (160716G)

Declaration by Supervisor

I/We have supervised and accepted this dissertation for the submission of the degree.

.....	.....
Dr. Subodha Charles	Date
.....	.....
Dr. Tharaka Samarasinghe	Date

Abstract

NON INTRUSIVE REAL TIME POWER MONITOR

Group members:D.E.T.I. Nanayakkara, P.S.S.B. Pathiranage, R.M.I.P. Rajapakshe, P.M.N.Wijewardena

Main Supervisor: Dr. Subodha Charles
Co Supervisor: Dr. Tharaka Samarasinghe

Keywords: Smart energy monitoring, Non intrusive load monitoring, Sequence to short sequence prediction, Energy disaggregation,

Non-intrusive load monitoring (NILM) which is also referred to as energy disaggregation is the problem of inferring the appliance level power consumption in a house given the aggregate power consumption. The objective of NILM is to reduce the electrical energy wastage in households by making users aware of power consumptions in appliance level. Although complete end to end commercial NILM solutions are available in the market, such a system customized to Sri Lankan households is not available. Hence, this report investigates the development of a complete end to end NILM system as a proof of concept, which is capable of sampling the aggregate power signal in a house using an active power monitor, inferring the power consumption of individual appliances given the aggregate power consumption using a deep learning based disaggregation algorithm deployed in the cloud and displaying the inferred appliance level power consumption to the user in real time using a web application.

We propose a novel attention based autoencoder architecture for energy disaggregation with enhanced real-time capability and accuracy. We developed the electronic active power monitor which is responsible for aggregate power sampling and transmission of power samples to the cloud. The disaggregation algorithm is customized to Sri Lankan households using the custom dataset collected from our active power monitor. We also developed the cloud based data pipeline which manages the overall data flow from the users home to the web front end. A user-friendly React web application is developed to display the disaggregation results to a user.

The disaggregation algorithm is tested on both the custom data and already available standard datasets. The high performance for custom data depicts that the performance of NILM algorithms can be improved by customizing the training to different regions and households. The results for standard datasets demonstrate the superiority of the proposed algorithm compared to the state of the art algorithms. Through the results, we also numerically evaluated the trade off between real-time capability and accuracy of NILM systems.

DEDICATION

To our parents and teachers

Table of Contents

List of Figures

List of Tables

Acronyms and Abbreviations

NILM	**N**on **I**ntrusive **L**oad **M**onitoring
SOTA	**S**tate **O**f **T**he **A**rt
ISR	**I**nterrupt **S**ervice **R**outine
SSID	**S**ervice **S**et **ID**entifier
HTTP	**H**yper **T**ext **T**ransfer **P**rotocol
MAE	**M**ean **A**bsolute **E**rror
MSE	**M**ean **S**quared **E**rror
MAPE	**M**ean **A**bsolute **P**ercentage **E**rror
SoC	**S**ystem **O**n a **C**hip

Chapter 1

INTRODUCTION

1.1. Problem identification

With the looming increase of energy usage over the years, the public awareness towards energy conservation has been expanding[2]. An attractive savings of energy lies in the residential sector which promises more than 12% reduction of consumption through consumers, with most of the savings arising through appliance specific real-time recommendations[3]. The total energy consumption of a user which is presented in the form of a bill at the end of the month does not provide information on how the cost can be reduced. More useful information can be provided through load monitoring where the contribution of each appliance to the total consumption is monitored. In addition, a psychological motivation to the cost reduction process can be introduced by displaying the power consumption of individual appliances to the user in real-time. Load monitoring has been an extensively researched topic. The monitoring can be done intrusively[4] or non-intrusively[5], where in the former case the appliances are monitored using dedicated smart meters and in the latter case the appliance level power consumption are inferred from the aggregate power consumption. Although Intrusive Load Monitoring is an ideal solution to give consumer feedback, the extensive hardware requirement imposes restrictions on it's deployment at a commercial scale. In contrast non-intrusive load monitoring (NILM) which is also referred to as energy disaggregation, requires much less hardware complexity, although efficient and powerful algorithms are required for the disaggregation task.

1.2. Scope

The types of electrical appliances used in different countries are different in many aspects. Hence, customizing NILM solutions to different regions can greatly improve the disaggregation accuracy. Although complete end to end commercial NILM solutions are available in the market, such a system customized to Sri Lankan households is not available. Therefore we developed a complete end to end NILM system as a

proof of concept, capable of sampling the aggregate power signal of the house and inferring the appliance level power consumption in real time. The aggregate power sampling is done using the active power monitor which then transmits the power samples to the cloud. The disaggregation algorithm deployed in the cloud infers the appliance level power consumption from the aggregate power samples. The data pipeline component is responsible for receiving the aggregate power samples transmitted by the active power monitor, presenting the samples to the disaggregation algorithm and storing the results in the database. The web application displays the disaggregation results and provides real time insights about the power consumption to the user.

1.3. Literature review

Hart in his work [5] elaborated on the importance of appliance signatures for the NILM task, and categorized them into steady state signatures and transient signatures. Transient signatures such as switch on/off transients[6], harmonics[7], electrical noise[8], VI trajectory curves are highly distinctive across appliances, which makes it a powerful tool for energy disaggregation[9]. Steady state signatures such as power difference between successive operation states, power consumed and the time spent in a particular power state are commonly used in the literature due to the low sampling rate and low computational complexity requirement although limitations such as not being able to identify a load that gets switched on and off within a sampling period and not being able to distinguish loads with similar steady state power consumption exist.

Various techniques have evolved over the years for energy disaggregation. Supervised learning approaches such as integer programming[10], genetic algorithms[11], knapsack algorithm based approaches, bayesian inference based approaches[12] and hidden markov models are exploited in the literature. Unsupervised and semi supervised learning methods such as motif mining[13], factorial hidden markov models[14] , unsupervised bayesian learning[15] and discriminative sparse coding[16] have also been used in the literature.

The latest areas of focus of NILM lie in the use of probabilistic graphical models such as factorial hidden markov models and deep learning based approaches. Due to the availability of computational power training deep neural networks with a large

number of parameters on large volumes of data is possible. Neural NILM[17] shows the superiority of deep neural network based approaches over the classical methods for the energy disaggregation task. Multilayer perceptrons, convolutional neural networks, long short term memory networks and attention based neural networks are exploited in the literature for the disaggregation task[17].

1.4. Method of investigation

Due to the superiority of deep learning based approaches for energy disaggregation we experimented several deep learning based models such as multilayer perceptrons, convolutional neural networks, convolutional autoencoders and attention based models from which we developed a novel attention based autoencoder architecture. We propose a methodology to enhance both the real-time capability and the accuracy of NILM systems by incorporating two modelling variations of the attention based autoencoder architecture. The data required for training the models were obtained using the active power monitor we developed. We experimented data preprocessing and data synthesis techniques to improve both the quality and quantity of training data prior to model training. The active power monitor development can be summarized under printed circuit board (PCB) design, PCB testing and firmware development. We also conducted a simulation of the active power monitor prior to development. Figure 1.1 displays design process of the three components, disaggregation algorithm, active power monitor and the cloud based components. Final step is the integration of the three components as shown in the figure. Notice that the active power monitor is used for data collection.

1.5. Principal results of the investigation

The disaggregation algorithm gave promising results when tested using custom data implying that the performance of NILM systems can be improved significantly by customizing to different regions and households. Moreover, the algorithm outperformed the state of the art energy disaggregation algorithms when tested using standard NILM datasets. The data synthesis and the data preprocessing techniques improved both the quality and quantity of training data which improved the model

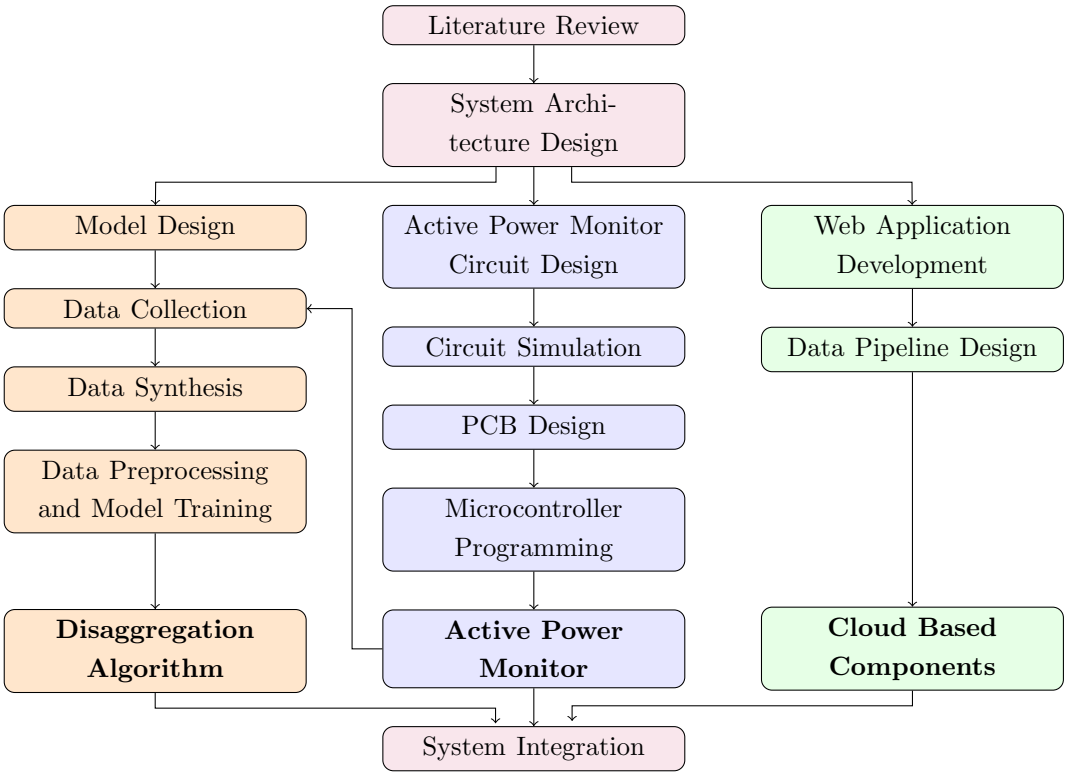


FIGURE 1.1: Method of investigation.

performance significantly. We also numerically evaluated the trade off between real-time capability and accuracy of NILM systems using the two modelling variations mentioned in Section 1.4. The active power monitor is capable of carrying out voltage and current waveform sampling at a 1kHz frequency and active power sampling at $\frac{1}{3}$ Hz and transmitting frames over to the cloud using a WiFi connection. The power measurement error was calculated to be around 3%. The web application we have developed provides an interface for the user to observe the inferred and processed results. Moreover, the user can obtain the daily and monthly appliance level energy consumption which are calculated using time scheduled cloud functions using the web application.

Chapter 2

METHOD

Different appliances have unique power consumption curves with distinct characteristics which are referred to as appliance signatures. As shown in Figure 2.1 the washing machine power consumption curve exhibits unique characteristics during the heating and spin cycles. Similarly as shown in Figure 2.2, the power consumption curve of a refrigerator consists of periodic lows and highs and sudden large surges during defrost mode. These signatures enable the idea of NILM. The disaggregation algorithms should have the capability to extract the signatures from the aggregate power signal in order to infer the appliance level power consumption.

Motivated by the above facts we developed a complete end to end NILM system as a proof of concept, comprising of following major components.

1. Disaggregation algorithm
2. Active power monitor
3. Data pipeline
4. Data presentation (Web application)

Figure 2.3 depicts the integration of the above components to form the complete NILM system. The active Power Monitor samples the aggregate active power signal from the user's home and transmits the samples to the cloud in near real time using a WiFi connection. The energy disaggregation algorithm takes the aggregate power samples as the input and infers the appliance level power consumption. The data pipeline manages the overall data flow from the users home to the web front end. The web application presents the inferences to the user. The remainder of the chapter focuses on describing the design process of the above components.

2.1. Developing the disaggregation algorithm

Let P denote the total power consumption of a house at any given time, such that it represents the aggregate of the power consumed by individual appliances in the

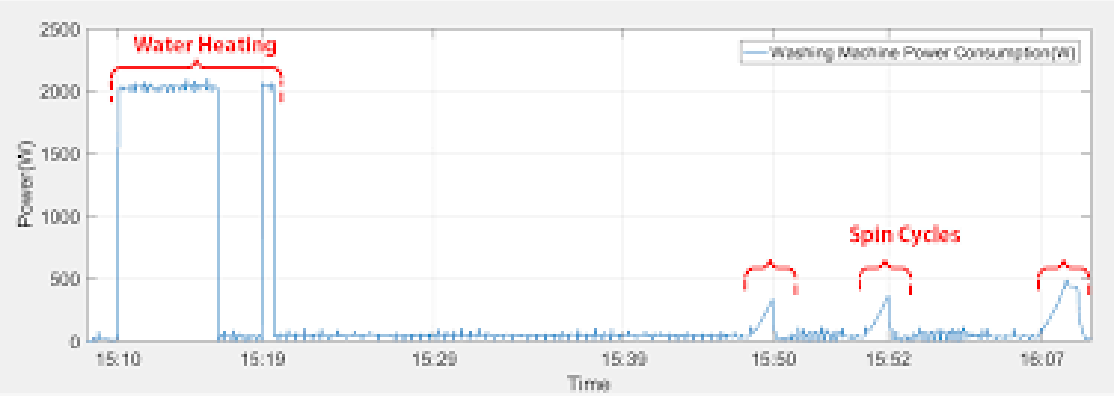


FIGURE 2.1: Signature of the washing machine (Source [1])

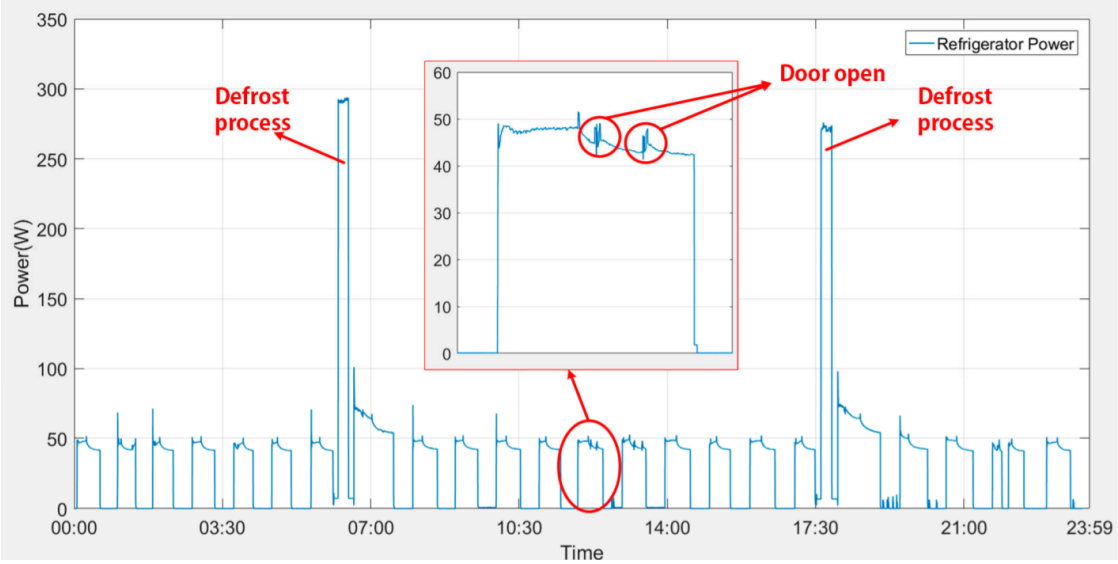


FIGURE 2.2: Signature of the refrigerator (Source [1])

household. Let p_i denote the power consumed by the i-th appliance. The NILM concept involves in finding each p_i without directly measuring it, through inference from the aggregate power consumption P.

More formally, let there be n power consuming appliances in the house. Let the aggregate power consumption at time t be given by $P(t)$. Similarly, if $p_i(t)(1 \leq i \leq n)$ and $e(t)$ represent the power consumption of the i-th appliance and the undesired noise, at time t, respectively, the aggregate power consumption can be written as

$$P(t) = \Big(\sum_{i=1}^{n} p_i(t) \Big) + e. \tag{2.1}$$

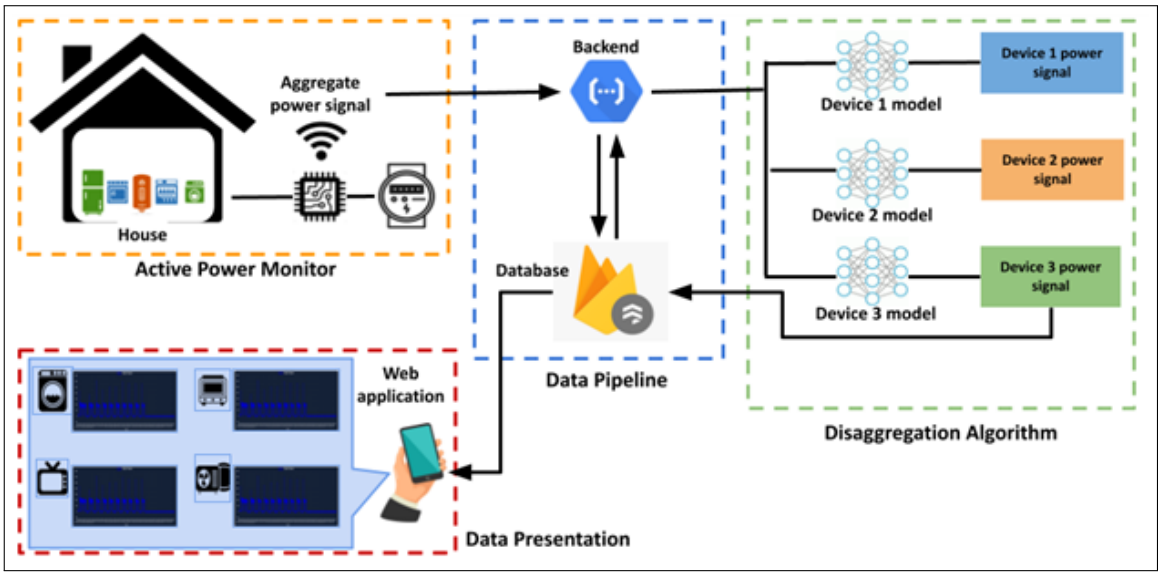


FIGURE 2.3: System block diagram

The smart electricity meter at the house samples signal $P(t)$ at intervals of T seconds, and outputs a series of samples $\hat{P}[j]$, for $j \in \mathbb{Z}$) and $\hat{P}[j] = P(jT)$. Similarly, the per-appliance samples are denoted by $\hat{p}_i[j] = p_i(jT)$. The NILM problem is to infer $\hat{p}_i[j]$ given $\hat{P}[j]$ for each $i \in [1, n]$.

We use a deep learning based algorithm for the NILM problem. First, we justify the use of deep learning for disaggregation and then we describe the methodology of developing the disaggregation algorithm under following phases.

- Model design
- Model training

Finally, we describe the inference process.

2.1.1. Why deep learning for NILM

Various techniques have been adopted in the literature for NILM. Recently, Kelly et al. [17], established the superiority of deep learning based methods for energy disaggregation compared to classical methods such as hidden Markov models. The major reason for this improvement is the feature extraction capability of deep learning models. Due to the presence of distinctive appliance signatures as described

previously, the feature extraction capabilities of deep learning methods will greatly improve the disaggregation performance. Next reason is the generalizability of deep learning based methods. The problem is addressed as a feature extraction problem which eliminates the limitations caused by the sum-matching approaches. Hence performance of the algorithm will be superior in the presence of previously unseen appliances. Finally, the deep learning based methods have the ability to deal with long range temporal dependencies, where the signatures which appeared over long durations can be effectively combined for prediction.

2.1.2. Model design

A dedicated model is used to infer the power consumption of each appliance given the aggregate power consumption. The appliances encountered in typical households can be categorized as multi-state appliances and appliances with continuously varying power demands. Washing machine, refrigerator and kettle are examples for multi-state appliances whereas lighting and laptop chargers can be categorized as appliances with continuously varying power demand. Since the appliance on time is both long and unpredictable for appliances of the second category, we only focus on modelling multi-state appliances. In fact majority of the household appliances can be categorized as multi-state appliances.

Deep neural networks suffer from vanishing and exploding gradient problems when the inputs to the model are long sequences. Hence, we adopt a sliding window based approach for modelling, where a window of the aggregate power sequence is input to the model from which a window of the appliance level power sequence is expected as output. Several sliding window methodologies are adopted in the literature for NILM. In "sequence-to-sequence learning", an appliance level power window corresponding to the input window is expected as the output [18], and in "sequence-to-point learning", a single point of the corresponding appliance level power window is expected as the output [19]. Sequence-to-point architectures have high computational complexity during inference. Sequence-to-sequence architectures generate a single output for the duration of the input window which is undesirable in real time implementations since the input window may span a long duration. Hence we adopt "sequence-to-short sequence learning" where a sub-sequence of an appliance level power window corresponding to the input window is expected as the

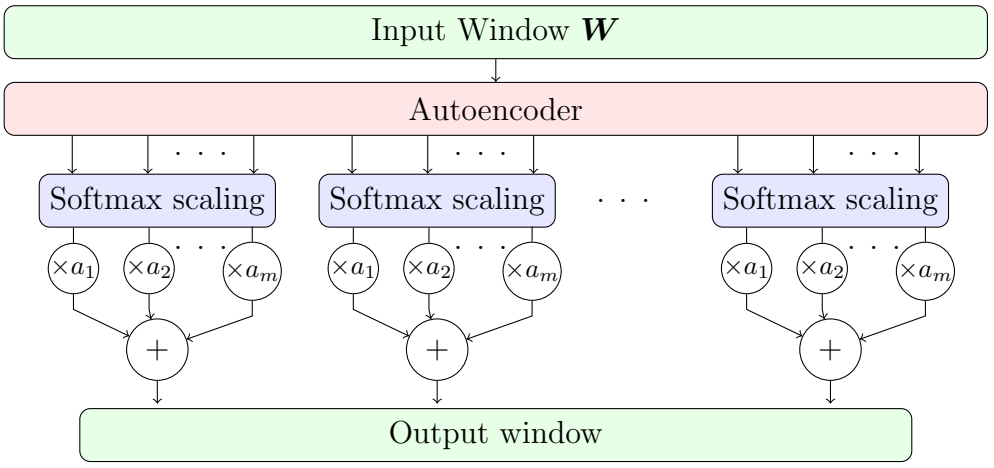


FIGURE 2.4: The attention based autoencoder architecture

output. Formally, the input to the model for the i^{th} appliance is $\hat{P}[j : j + w_i]$, where $j \in [1, l - w_i + 1]$, l is the length of the power sequences and w_i is the input window length of the i^{th} appliance. The expected output from the model is $\hat{p}_i[j + s_i : j + s_i + \hat{w}_i]$, where $\hat{w}_i \leq w_i$ is the output window length of the i^{th} appliance and $s_i \in [0, w_i - \hat{w}_i]$.

Note that the time gap between sampling of the power value corresponding to the first timestamp of the output window and the generation of the model output is $(w_i - s_i)T + T'$ where T' is the time taken for inference and other processing tasks. Hence, increasing s_i is favourable for a real-time system. On the other hand, when $w \gg \hat{w}$ and s_i is close to $w_i/2$, the input power window captures a significant amount timestamps before and after each timestamp of the output window leading to superior disaggregation performance. Hence, we propose two variations of sequence-to-short sequence learning, where in the first variation we use s_i close to $w/2$ and in the second variation we use s_i close to w . It should be noted that both variations are implemented using the attention based autoencoder architecture depicted in Figure 2.4. The architecture consists of the output soft-max scaling layers and the autoencoder which are explained below.

2.1.2.1. Soft-max scaling layers

Consider an appliance with m states and let the mean power consumption of the u^{th} state be a_u. Let r_t be the random variable denoting the power value at t^{th} $(1 \leq t \leq \hat{w})$ timestamp of the output window, where $\hat{w}$ is the output window size.

Then the expected value of r_t given the aggregate window can be written as,

$$\mathbb{E}\{r_t|\boldsymbol{W}\} = \sum_{u=1}^{m} \Pr(s_t = u|\boldsymbol{W})a_u, \tag{2.2}$$

where s_t is the random variable representing the state of the appliance at the t^{th} timestamp of the output window and $\boldsymbol{W}$ represents the aggregate power window of length w. The model predicts the sequence of random variables s_t for $1 \leq t \leq \hat{w}$. Formally, the model infers the matrix $\boldsymbol{M}$, where $M_{t,u} = \Pr(s_t = u|\boldsymbol{W})$ from which the output window can be computed using Equation 2.2. To model $\boldsymbol{M}$, a soft-max scaling layers are added at the model output. The output window is predicted using a feed-forward layer as shown in Figure 2.4.

The modelling can be done either by training the model to predict $\boldsymbol{M}$ using categorical cross entropy as the loss function or by training the model to predict the output window directly using mean squared error as the loss function. In the former case, state distribution for the appliance level power values in the dataset and the mean power values a_u are obtained using Gaussian mixture models and expectation maximization algorithm and in the latter case, they are learnt during model training. We only present the second approach since both gave similar results in our exploration.

2.1.2.2. Autoencoder

The autoencoder depicted in Figure 2.5 was designed after evaluating several models such as, convolutional neural networks (CNN), recurrent neural networks (RNN), residual networks, attention based models. The model architecture is similar to the transformer model presented in [20]. The model consists of an encoder with convolutional layers and a decoder with convolutional, multi-head attention and LSTM layers. The convolutional encoder extracts the signatures of the appliances whereas the multi-head attention and the LSTM layers in the decoder filter out the irrelevant signatures.

The outputs from all the convolutional layers are zero padded in order ensure equal dimensions for the input and the output of the layer. Moreover, the long short term memory layers are set to output the state vector at all time-steps. The architecture of the multi-head attention layer is shown in Figure 2.7. The inputs to the multi-head attention layer are the three matrices $\boldsymbol{Q}, \boldsymbol{V}, \boldsymbol{K}$ of dimension $\hat{w} \times 32$. The input

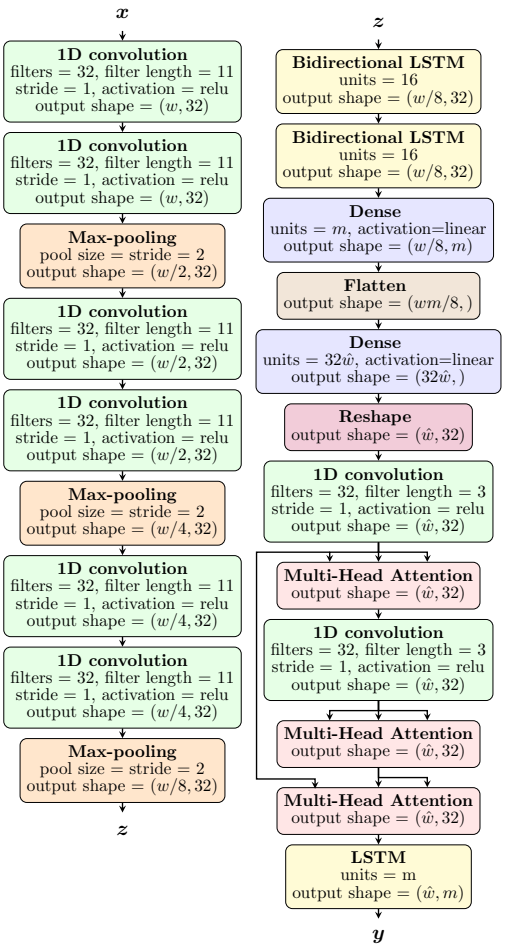


FIGURE 2.5: Autoencoder

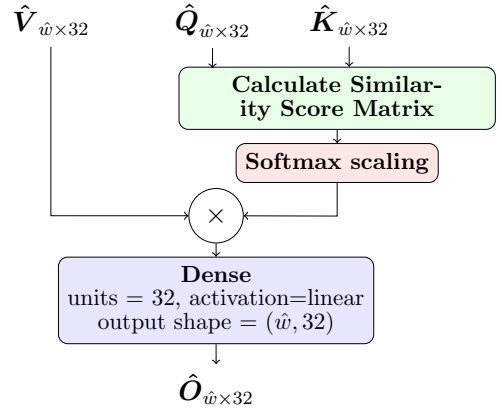


FIGURE 2.6: Attention Layer

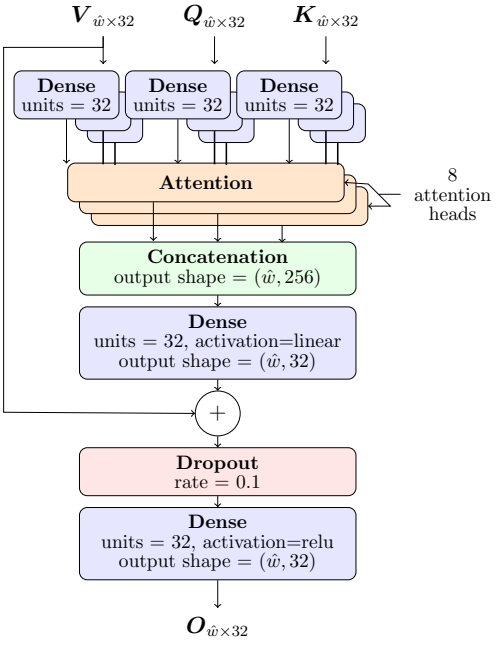


FIGURE 2.7: Multi-Head Attention

matrices are sent through independent dense layers to obtain h triplets of equal dimensional matrices ($\hat{\boldsymbol{Q}}$, $\hat{\boldsymbol{V}}$, $\hat{\boldsymbol{K}}$) as shown in Figure 2.7. Each of the triplets are input to independent attention layers, from which the obtained outputs are processed as shown in Figure 2.7 to obtain the layer output $\boldsymbol{O}$ having the same shape as the inputs. An attention layer maps a query and a set of key value pairs to an output [20]. Formally, the attention layer takes three equal dimensional matrices $\hat{\boldsymbol{Q}}$,$\hat{\boldsymbol{V}}$ and $\hat{\boldsymbol{K}}$ of shape $\hat{w}\times 32$ as input, and maps them to an output matrix $\hat{\boldsymbol{O}}$ as shown in Figure 2.6. The rows of $\hat{\boldsymbol{Q}}$ given by $\boldsymbol{q}_i$ $(1 \leq i \leq w)$ are the queries. The values $\boldsymbol{v}_i$ and the keys $\boldsymbol{k}_i$ are defined similarly. The first step of attention is calculating the similarity score

matrix $\boldsymbol{S}$ between the $\hat{\boldsymbol{Q}}$ and $\hat{\boldsymbol{K}}$. The entry $S_{i,j}$ is defined to be the similarity score between $\boldsymbol{q}_i$ and $\boldsymbol{k}_j$. Different functions have been used to calculate the similarity score in existing literature [21]. We use $S_{i,j} = \boldsymbol{d_a}^\top \tanh(\boldsymbol{W}_q \boldsymbol{q}_i + \boldsymbol{W}_k \boldsymbol{k}_j + \boldsymbol{b})$, where $\boldsymbol{W}_q, \boldsymbol{W}_k$ are learnt matrices and $\boldsymbol{d_a}, \boldsymbol{b}$ are learnt vectors. In our model, we let $\boldsymbol{W}_q, \boldsymbol{W}_k \in \mathbb{R}^{4\times 32}$ and $\boldsymbol{d_a}, \boldsymbol{b} \in \mathbb{R}^4$. Secondly, the entries of matrix $\boldsymbol{S}$ are normalized such that the entries along a row sum to 1. Formally, we let

$$\hat{S}_{i,j} = \frac{\exp(S_{i,j})}{\sum_{p=1}^{W} \exp(S_{i,p})}, \tag{2.3}$$

where $\hat{\boldsymbol{S}}$ represents the normalized matrix. Finally, the product $\hat{\boldsymbol{S}}\hat{\boldsymbol{V}}$ is sent through a dense layer to obtain $\hat{\boldsymbol{O}}$.

2.1.3. Model training

The model training phase can be summarized under three main stages,

- Data collection
- Data synthesis
- Data preprocessing and training

First the data requirement for model training is obtained. The available data is synthesized in order to improve both the quality and the quantity of data. The data is then pre-processed. Finally the models are trained using the data.

2.1.3.1. Data collection

Data is a major requirement in modern day deep learning. Data collection is done by periodically sampling the aggregate power and the appliance level power using the active power monitor. Then the dataset is prepared. The dataset consists of simultaneous readings of the aggregate power signal and separate appliances' power signals with a timestamp.

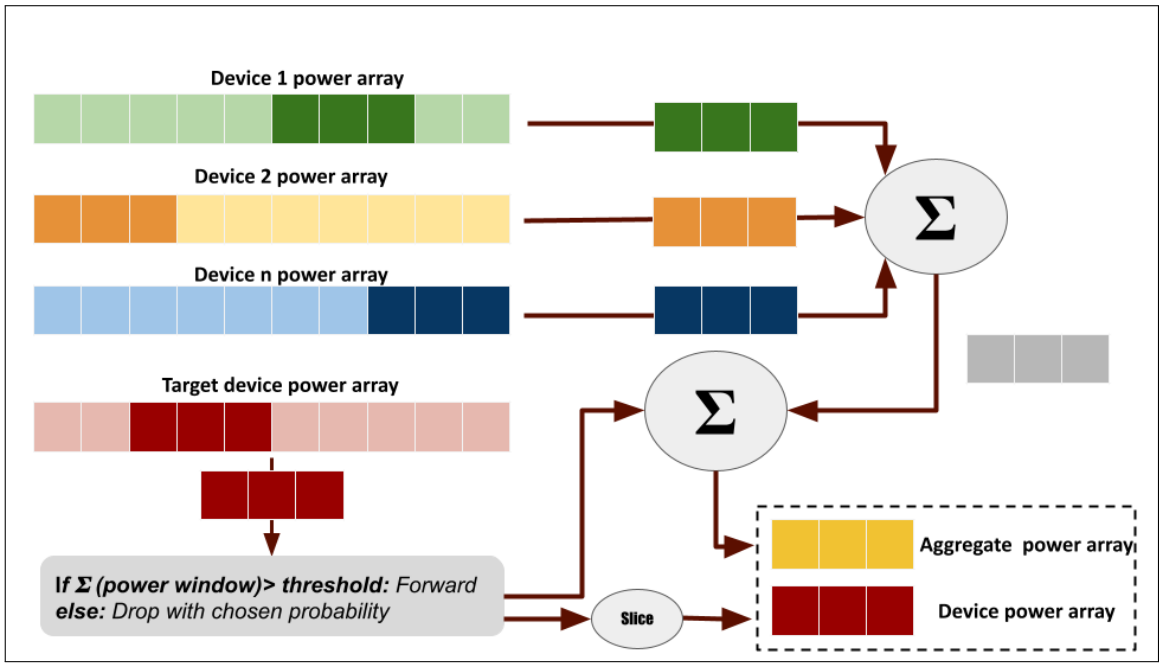


FIGURE 2.8: The data synthesis technique

2.1.3.2. Data synthesis

We adopt a novel data synthesis technique where training data is synthesized from the dataset obtained in the data collection phase. Data synthesis helps to increase the quantity of training data as well as to improve the generalizability of the models. The data synthesis process is outlined in Algorithm 1 and is illustrated graphically in Figure 2.8. We consider the dataset of aggregate power windows from a given house to be X. The dataset of appliance level power windows of the i^{th} $(1 \leq i \leq N)$ appliance from the same house is denoted by Y. Both datasets which are initially empty, are filled using the synthesized data windows during data synthesis. The data synthesis for the other appliances and houses follow the same procedure.

The appliance level power windows are generated by sliding a window of length w_i through $\hat{p}_i$ with a shift $\hat{w}_i$ between two consecutive windows. We first obtain the appliance level power window $\boldsymbol{y}$ of length w_i (line 5). If $\boldsymbol{y}$ contains at most P_i^{num} values greater than P_i^{thresh} we discard $\boldsymbol{y}$ with a probability $1 - P_i^{\text{choose}}$ (lines 6-14). If $\boldsymbol{y}$ is not discarded, we generate c aggregate power windows corresponding to $\boldsymbol{y}$ (line 4). An aggregate power window $\boldsymbol{x}$ corresponding to $\boldsymbol{y}$ is generated by adding power windows from each appliance except the i^{th} appliance to $\boldsymbol{y}$. Formally, we generate

$\boldsymbol{x}$ by letting

$$\boldsymbol{x} = \boldsymbol{y} + \sum_{\substack{k=1 \\ k \neq i}}^{\hat{N}} \hat{p}_k[b_k : b_k + w_i], \tag{2.4}$$

where b_k is chosen randomly such that, $1 \leq b_k \leq l - w_i + 1$ (line 15-21). Then we add $\boldsymbol{x}$ and $\boldsymbol{y}[s_i : s_i + \hat{w}_i]$ to X and Y respectively for each $\boldsymbol{x}$ (line 22). The selective discarding of appliance level power windows prevents the model from learning to predict flat power signals due to the skewed data distribution of some appliances which is a result of the appliance being in low power consumption modes or being turned off for most of the time.

Note that time dependencies and usage patterns between appliances will be lost due to the data synthesis technique. Although appliance usage patterns and usage time (time of the day) can improve performance in the presence of common appliance combinations such as toaster and kettle as exploited by [22, 23], the performance in the presence of rare appliance combinations will be reduced, which will reduce the generalizability of the model.

2.1.3.3. Data pre-processing and training

During data pre-processing, the model input and target values are scaled linearly such that they lie in $[0, 1]$. Appropriate scaling improves the learning ability of the models by avoiding vanishing and exploding gradient problems.

Some devices may have instantaneous sharp peaks of power which may be due to measurement noise or switching transients. Scaling the data in the presence of these peaks, will lead to very small values of the other sections of the power signal. Hence, we use median filtering as a noise removal step prior to scaling.

Finally, the obtained data is used for model training The data is fed in batches of shape (b, w) where b, w are the batch size and the window size respectively. We adopted several metrics such as mean squared error (MSE), mean absolute arrow(MAE) and mean absolute percentage error (MAPE) out of which MSE gave

Algorithm 1: Algorithm for training data synthesis for the i^{th} appliance in a given house.

Data: $\hat{p}_k$ for $1 \leq k \leq N$, P_i^{thresh}, P^{num}, P_i^{choose}, s_i, c
Result: X, Y

Set l to be the length of $\hat{p}_i$;
for $0 \leq n < (l - w_i)/\hat{w}_i$ **do**
  Set $f = (\hat{w}_i n)$;
  for $0 \leq j < c$ **do**
    Initialize $\boldsymbol{y} = \hat{p}_i[f : f + w_i]$, selected=False;
    Calculate number of values of $\boldsymbol{y}$ greater than P^{thresh} and let this value be g;
    if $g < P^{\text{num}}$ **then**
      Sample $r \in U(0, 1)$;
      if $r > P_i^{\text{choose}}$ **then**
        Set selected = True;
      end
    else
      Set selected = True;
    end
    if selected *is* True **then**
      Initialize $\boldsymbol{x} = \boldsymbol{y}$;
      for $1 \leq k \leq \hat{N}$ and $k \neq i$ **do**
        Sample $b_k \in [1, l - w_i + 1]$;
        Add $\hat{p}_k[b_k : b_k + w_i]$ to $\boldsymbol{x}$;
      end
    end
    Add $\boldsymbol{x}$, $\boldsymbol{y}[s_i : s_i + \hat{w}_i]$ to X,Y;
  end
end

the best results. We used Adam and stochastic gradient descent optimization algorithms from which Adam optimizer led to better results with better generalization capability. We used tensorflow 2.5 as the framework for model implementations.

2.1.4. Inference

During inference, the input window is preprocessed by median filtering and min-max scaling similar to the training phase. The scaling parameters learnt during training are used for min-max scaling. The inferred output window is re-scaled using the respective scaling parameters learned during model training.

The disaggregation results are given in Section 3.

2.2. Developing the active power monitor

Active Power Monitor is the device which is used to sample the aggregate active power consumption of the user's home and transmit the power samples to the cloud via WiFi. The active power monitor is also used for data collection described in Section 2.1.3.1. The development of the active power monitor can be summarized under following major stages.

- Circuit design
- Simulation and verification
- Micro-controller programming
- Circuit calibration

The remainder of the Section 2.2 describes each stage in detail.

2.2.1. Circuit Design

The instantaneous voltage and current values required to calculate the active power is measured using a voltage transformer and a clip on current transformer respectively. The measured quantities are scaled and shifted prior to sampling using the analog to digital converter of the microcontroller. In addition to the above features the circuit is responsible for positive and negative voltage supply generation, reference voltage generation and micro-controller stabilization. Considering the above requirements, the circuit is designed to have the following major sections.

- Power supply section
- Signal scaling and biasing section
- Micro-controller section

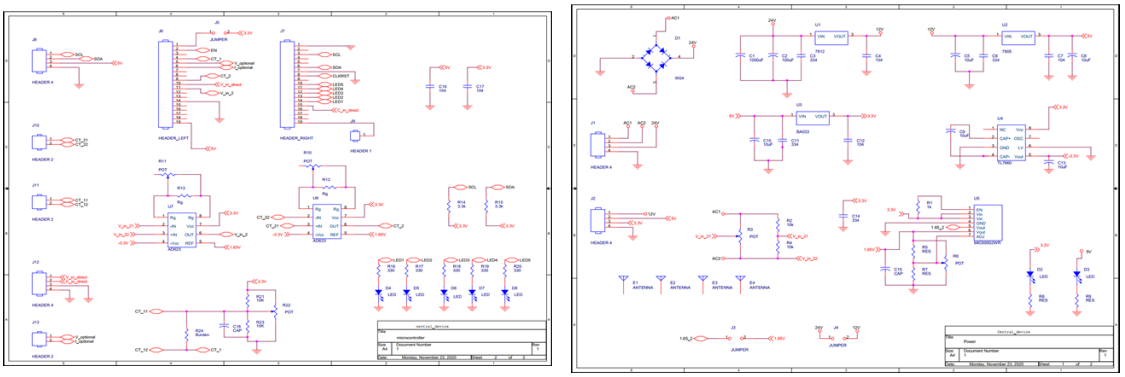


FIGURE 2.9: Schematics of the circuit

Each section is described below in detail. The PCB of electronic circuit is designed using Orcad circuit design software tool. The schematics of the circuit are shown in Figure 2.9. The CAD design for the PCB is shown in figure 2.10 and completed PCB is shown in figure 2.11.

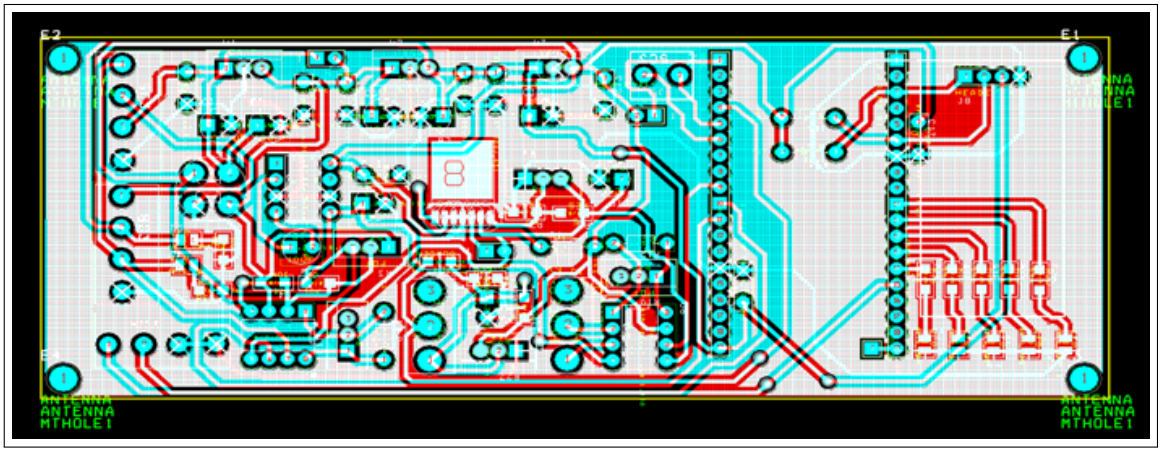


FIGURE 2.10: CAD design of the PCB

2.2.1.1. Power supply section

Power supply section powers the active power monitor. The circuit is designed to be powered using a 220V - 12V step down transformer. The AC voltage is rectified using a rectifier IC (WD 04) and smoothed using smoothing capacitors to produce a DC voltage of 13V. Then voltage is then regulated using regulators of output voltage 5V and 3.3V. Necessary smoothing capacitors are included in the circuit.

FIGURE 2.11: Soldered and completed PCB

2.2.1.2. Signal scaling and biasing section

This section is responsible for pre-scaling and shifting the current and voltage measurements obtained from the voltage and the current transformers prior to sampling using the microcontroller. The current transformer generates a voltage waveform proportional to current value to be measured. This waveform oscillates around 0V with a small peak to peak voltage. We amplify and shift this voltage waveform to the range of 0-3.3V using an instrumentation amplifier so that the micro-controller can sample with a larger resolution. The signal generated by the voltage transformer spans a larger voltage range compared to the range acceptable by the micro-controller. Therefore we first use a potential divider to reduce the range and then use an instrumentation amplifier to shift the signal. The use of an instrumentation amplifier also adds a denoising effect.

2.2.1.3. Micro-controller section

The micro-controller section is responsible for sampling the active power and transmitting active power samples to the cloud. The analog to digital converter of the

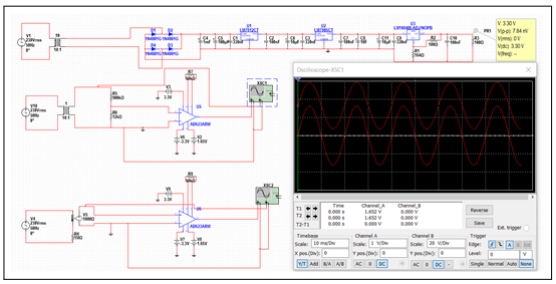


FIGURE 2.12: The voltage sampling circuit simulation

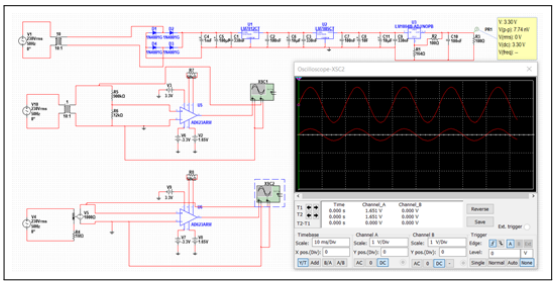


FIGURE 2.13: The current sampling circuit simulation

micro-controller samples analog signal inputs from signal amplification and biasing section described in Section 2.2.1.2. The algorithm implemented in the micro-controller calculates the active power samples which are then sent to the cloud via WiFi for processing. We selected ESP32 Dev kit 4 which has inbuilt WiFi support as the micro-controller solution. The micro-controller is powered using 5V DC power. Details about microcontroller programming will be discussed in section 2.2.3.

2.2.2. Simulation and verification

The designed circuit was simulated and verified using Multisim prior to implementation. Voltage sampling and current sampling stages were simulated as shown in Figure 2.12 and Figure 2.13.

2.2.3. Micro-controller programming

The ESP32 module mentioned in Section 2.2.1.3 has inbuilt WiFi capabilities along with two microcontroller cores. The micro-controller is programmed using C language. A higher level code base is available in the ESP-IDF coding environment which is built to program ESP32 modules to access the inbuilt resources such as WiFi. The micro-controller program consists of the following major components.

- Active power sampling algorithm
- Data transmission over WiFi

First, we describe the active power sampling algorithm followed by the data transmission process over WiFi. The above two tasks are distributed among the two cores of the microcontroller as described in Section 2.2.3.3.

2.2.3.1. Active power sampling algorithm

The active power sampling algorithm first samples the voltage and current values in high frequency from which the low frequency active power samples are calculated. The active power sampling process is outlined in Algorithm 2.

First we sample the voltage and current with 1kHz frequency to obtain instantaneous voltage V_i and instantaneous current I_i samples (1 such sample is multi-sampled 8 times to reduce reading errors as shown in line 3 of Algorithm 2). We then obtain 3000 such V_i, I_i samples and calculate a single active power sample

$$P_a = \frac{\sum_{i=1}^{3000} V_i I_i}{3000}, \tag{2.5}$$

as shown in line 6. We obtain 8 consecutive power samples P_a and construct the active power sample frame (A_w)

$$A_w = [P_{a_1}, P_{a_1+1}, \cdots, P_{a_1+7}], \tag{2.6}$$

as shown in line 8. Finally, A_w is transmitted along with the UTC timestamp over WiFi to the cloud as described in Section 2.2.3.2. Observe that, active power samples are calculated with a frequency of $\frac{1}{3}$ Hz and active power frames are transmitted to the cloud with a frequency of $\frac{1}{24}$ Hz.

2.2.3.2. Data transmission over WiFi

We use the Home WiFi connection of the user to transmit active power data collected from active power monitor to the cloud. The ESP32 module connects with the WiFi router using the user's SSID and the password. The data transmission is handled by the core 1 of the microcontroller. The connection is established every 24 seconds and destroyed on the success of transmission. The microcontroller sends an HTTPS POST request which is structured as shown in Figure 2.14.

Algorithm 2: Active power sampling algorithm

Initialize L, A_w to be empty lists;
for each 0.001 second **do**
    Sample $V_i = (1/8)\sum_{j=1}^{8} \hat{V}_i$ and $I_i = (1/8)\sum_{j=1}^{8} \hat{I}_i$ where $\hat{V}_i, \hat{I}_i$ are 8 kHz samples;
    Add (V_i, I_i) to L;
    if L has 3000 samples **then**
        Calculate $P_a = (1/3000)\sum_{(V_i,I_i)\in L} V_i I_i$;
        Empty L;
        Add P_a to A_w;
        if A_w has 8 samples **then**
            Transmit A_w with the UTC timestamp to the cloud as described in Section 2.2.3.2;
            Empty A_w;
        end
    end
end

Startline	URL : CloudFunction1 URL	Method : POST
Header	Various Key value pairs for authentication	
Body	"Data={ProductID = XX12Y , Active Power Array = {sample1,....,sample8} , Timestamp={20:04:1….}}"	

FIGURE 2.14: Structure of the HTTP POST request

2.2.3.3. Work division between cores

Since the high frequency voltage and current sampling has to be carried out in exactly 1kHz frequency without being disturbed from other operations of the microcontroller, we created a timer interrupt service routine for the high frequency sampling function that triggers the interrupt action in 1kHz. With this timer ISR, we identified that the core gets overloaded when both the ISR and WiFi transmission process is assigned to a single core. Therefore we divided the work between two cores as shown in table 2.1.

Core 0 function	**Core 1 function**
Major task : Active Power Sampling Process	Major task : Transmission over WiFi process
High Frequency V,I sampling(1kHz)	Establishment and termination of WiFi connection
Low frequency Active Power Sampling(1/3Hz)	Retrieval of UTC timestamp
Building active power frame and handing over the frame to Core 1	Transmission of Active Power Array and timestamp using an HTTPs POST request

TABLE 2.1: Division of work between microcntoller cores

2.2.4. Circuit calibration

The microcontroller ADC requires a positive input signal not exceeding 3.3V. In order to ensure that the current samples and voltage samples meet this criteria, the voltage and current circuit calibration was executed.

The voltage signal received from the secondary of the voltage transformer V_{sig} is further stepped down using a potential divider. The resulting signal V_{in} is fed to an instrumentation amplifier IC as described in Section 2.2.1.2. The output of the voltage amplification IC is V_{out} where,

$$V_{out} = G_v V_{in} + C_v, \tag{2.7}$$

where G_v and C_v are constants to be determined and

$$V_{in} = \frac{V_{sig} R_1}{R_1 + R_2}, \tag{2.8}$$

where R_1 and R_2 are potential divider resistances. In voltage calibration we obtain (V_{in}, V_{out}) pairs for several device combinations from which we obtain the least squares fit for G_v and C_v.

The current transformer produces a voltage waveform proportional to the AC current flowing through the wire it has been clipped to. We amplify and shift this voltage waveform using an instrumentation amplifier IC as described in Section 2.2.1.2. Unlike the voltage circuit calibration, this circuit can only be calibrated using AC currents. The voltage produced by the current transformer V_{cin} is proportional to

the measured current A_{sig}. Therefore the output of the instrumentation amplifier IC is given by

$$V_{out} = G_c V_{cin} + C_c, \tag{2.9}$$

where G_c and C_c are constants to be determined. We obtain (V_{cin}, V_{out}) pairs for several device combinations with known current consumption after which we obtain the least squares fit for G_c and C_c. The results obtained for calibration are added in Section 3.2.

2.3. Data pipeline development

Figure 2.15 shows the data pipeline component which is responsible for receiving the active power samples (in the form of power frames as described in Section 2.2.3.1) transmitted by the active power monitor, presenting the power samples to the deep learning models and storing the results in a database. The data pipeline consists of the basic components which are required for end to end functionality of the system and the components which are added to enhance user experience. We describe each component in detail in the remainder of the section.

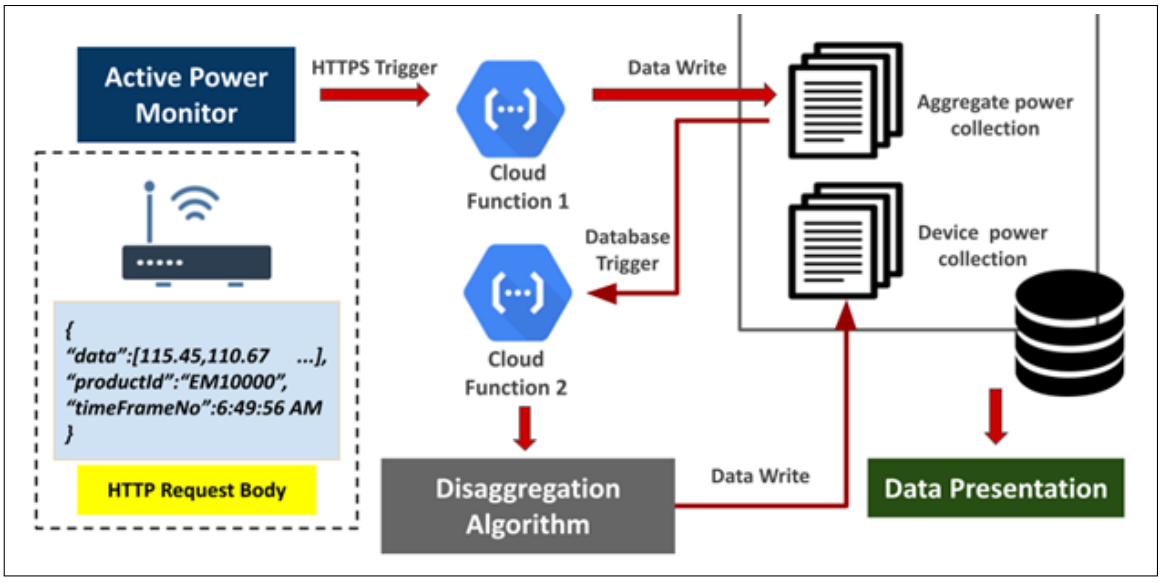


FIGURE 2.15: The data pipeline

2.3.1. Basic components of the data pipeline

The data pipeline is implemented using realization of the following basic components.

- Cloud function 1
- Aggregate power collection
- Cloud function 2
- Device power collection
- Data Presentation

The components are described in the order in which they are encountered by a power frame. We use services from Google Cloud Platform to implement the data pipeline. We describe each component in detail below.

2.3.1.1. Cloud function 1

Cloud function 1 is deployed to carry out following functions.

- Receiving the data transmitted by the electronic power monitor in the form of a HTTPS post request. The active power array, timestamp and the product id are present in the HTTPS message body as shown in Figure 2.14.
- Writing the active power array along with the timestamp and product id received from the HTTPS message to the aggregate power collection of the database described in Section 2.3.1.2 in the form of a document.

The function is written using JavaScript language and the run-time environment is Node.js 10. The trigger event of the cloud function is configured to be a HTTPS request.

2.3.1.2. Aggregate power collection

The aggregate power collection is where the aggregate power arrays are stored in the database. The aggregate power data stored here are utilized to infer the appliance level power consumptions and to display the total energy consumption to the user using the web application.

2.3.1.3. Cloud function 2

The cloud function 2 is triggered when a new document is written to the aggregate power collection by the cloud function 1. Cloud function 2 performs the following actions.

- The new document which triggered the cloud function 2 is extracted and forwarded to the energy disaggregation algorithm
- Writing the data back to the device power collection described in Section 2.3.1.4 as a document, when the inference results are returned by the disaggregation algorithm.

2.3.1.4. Device power collection

Device power collection is where the appliance level power consumption inferences are stored in the database. The inference data is written to this collection by the cloud function 2 as described in Section 2.3.1.3. The web application uses the data stored in this collection to display appliance level power consumptions to the user.

2.3.1.5. Data Presentation

This component is the end of the data pipeline. The data presentation component is the web application which displays the aggregate power data and appliance level power consumptions to the user. The web application development is described in Section 2.4.

2.3.2. Components added to enhance user experience

We now describe the the components which are added to enhance user experience. These components can be categorized under cloud functions and database collections.

2.3.2.1. Cloud Functions

Apart from the cloud functions described under the basic components, we have used time scheduled cloud functions in the backend of the web application, for calculating monthly and daily energy consumption values of the users.

2.3.2.2. Database Collections

Apart from the aggregate power and device power collections, we have several other collections in our database to serve the user. These databases are Firestore document collections with a structure imposed using a Schema.

- Issued products collection - power monitor units that have been issued to the user .
- User details collection - stores the user details.
- Daily aggregate energy collection - aggregate energy consumption data on a daily basis
- Daily device energy collection - device level energy consumption data on a daily basis
- Monthly aggregate energy collection - aggregate energy consumption data on a monthly basis
- Monthly device energy collection - device level energy consumption data on a monthly basis

2.4. Web application development

The web application is the data presentation component of our end to end NILM product. We have developed the web application using the react front end library.

The major task of the web application is presenting the real time aggregate and appliance level power consumption curves to the user. In addition, daily and monthly aggregate and appliance level power consumptions and the total energy consumptions in last month and the last week are presented. The web application also

FIGURE 2.16: Home page

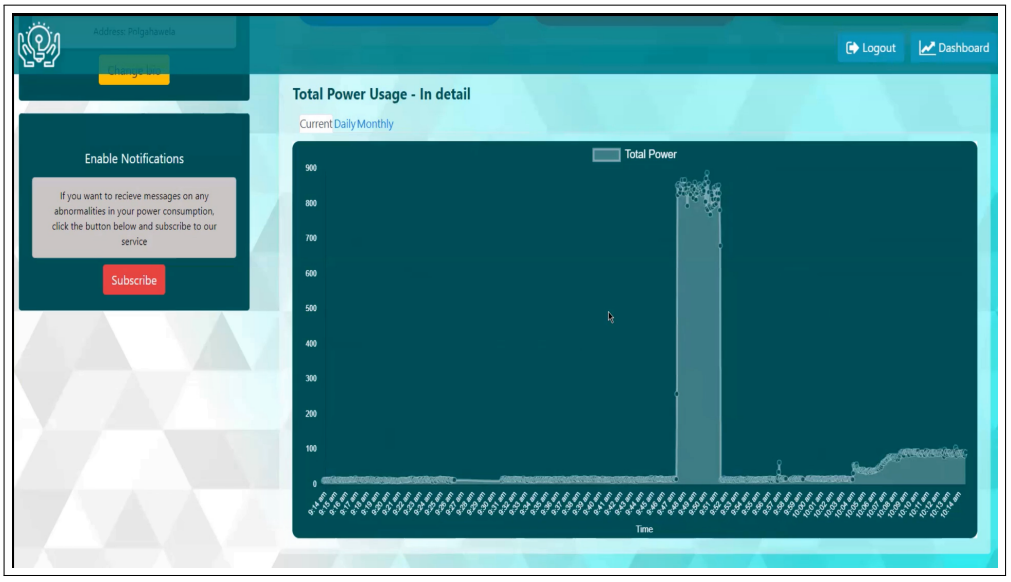


FIGURE 2.17: Power waveforms

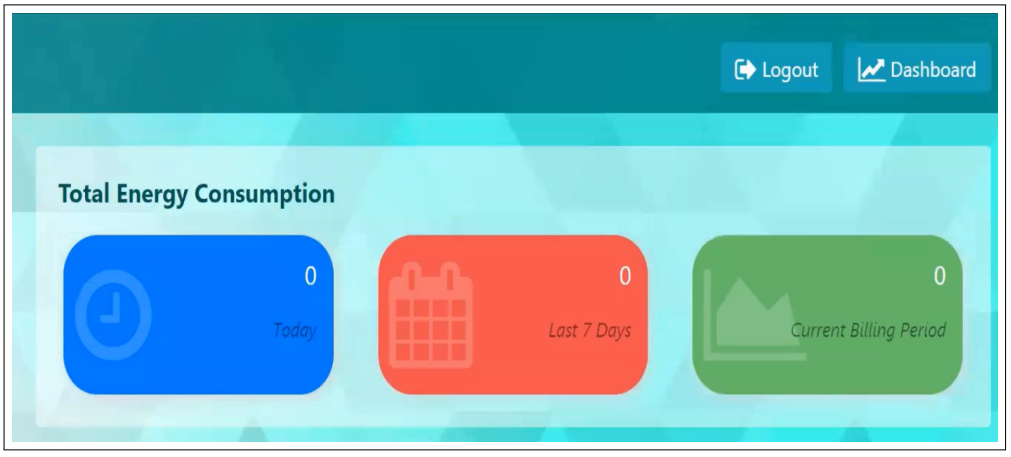


FIGURE 2.18: Power consumption data

displays the highest power consuming devices for last month, last week and today. Moreover, the web application is equipped with sign up, sign in and product id verification functionalities. The front end of the web application is shown in Figure 2.16, Figure 2.17 and Figure 2.18.

The redux architecture is used in the web application for state management such as login statuses in a single instance. We use functional components to present the view. The graphs in the web application are real time updated using a listener function of Firebase SDK. When aggregate and inferred power data are written to the database, the listener function sends the updated data to the web application and the graphs are updated using this new data. Lifecycle methods are used to manage the event flow in some occasions (e.g. real time listeners).

Chapter 3

RESULTS

In this chapter we provide the major results derived through our project. We first provide the results generated from the disaggregation algorithm. Then we provide the results for circuit calibration described in Section 2.2.4. Finally we provide the specifications of the active power monitor.

3.1. Energy disaggregation results

The number of states m (introduced in Section 2.1.2.1) and the appliance switch on power threshold p_{on}, P_i^{choose} and P_i^{num} (introduced in Section 2.1.3.2) for different devices are given in Table 3.1. P_i^{thresh} (introduced in Section 2.1.3.2) is set equal to p_{on}. The parameters m, P_i^{choose} and P_i^{num} are set by experimenting with several values. We consider appliance independent values for the parameters w_i, $\hat{w}_i$ and s_i (introduced in Section 2.1.2). Hence we denote the above parameters using w, $\hat{w}$ and s. We set w and $\hat{w}$ to 1024 and 8 respectively. We implement the two variations of sequence to short sequence learning described in Section 2.1.2 by letting $s = 508$ and $s = 1008$.

TABLE 3.1: The parameters used for different devices

	m	p_{on}	P_i^{num}	P_i^{choose}
Refrigerator	4	50	10	1
	4	50	20	0.5
Microwave	3	200	10	0.6
Kettle	4	300	10	0.6
Dish washer	4	50	10	0.5

We use the disaggregation performance metrics mean absolute error (MAE) and signal aggregate error (SAE) as the main evaluation metrics where

$$\text{MAE} = \frac{1}{T}\sum_{k=1}^{T}|\hat{y}_k - y_k|, \tag{3.1}$$

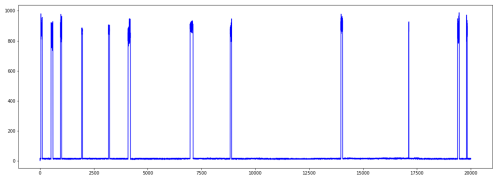


FIGURE 3.1: Data collected from kettle

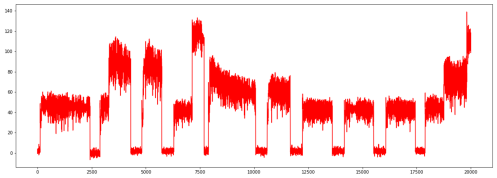


FIGURE 3.2: Data collected from refrigerator

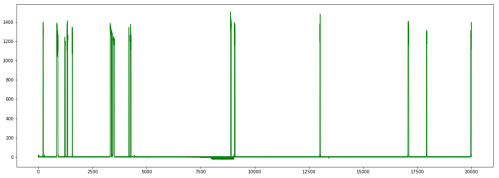


FIGURE 3.3: Data collected from microwave

$$\text{SAE} = \frac{|\sum_{k=1}^{T} \hat{y}_k - \sum_{k=1}^{T} y_k|}{\sum_{k=1}^{T} y_k}, \tag{3.2}$$

y_k and $\hat{y}_k$ are the actual appliance power consumption and the predicted appliance power consumption at time k [19]. MAE measures how well the model approximates the actual appliance power waveform, whereas the SAE measures how well the model approximates the total power consumption of a appliance over a period of time. Although in a NILM setting, MAE is a more intuitive metric, SAE will be useful from a user point-of view, since user will also care about the energy consumed by the appliance over a period of time. We also evaluate the models on event detection metrics such as precision, recall, accuracy and the F1 score, calculated using p_{on}.

The data collected from microwave, kettle and refrigerator over 20 hours are shown is Figures 3.1, 3.2 and 3.3 respectively. We use the shown data to train, test and validate our disaggregation algorithm.

Shown in Table 3.2 are the disaggregation results. It must be noted that the training and the testing data are obtained from the same house. Hence high event detection performance can be observed. We initially tested the models using both the modelling variations introduced in Section 2.1.2. Both variations correctly classified the devices as depicted by the high F1 score. Hence, we only provide the results for the second variation which has superior real-time performance as described in Section 2.1.2.

TABLE 3.2: The disaggregation results for kettle, refrigerator and microwave

Metric	**Kettle**	**Refrigerator**	**Microwave**
MAE	3.64	6.87	42.82
MNE	0.049	0.042	0.957
Recall	0.967	0.997	0.981
Precision	1.000	0.991	0.967
Accuracy	0.999	0.990	0.998
F1 Score	0.983	0.994	0.973

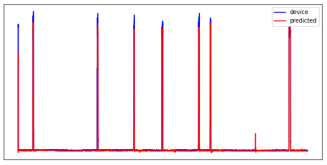


FIGURE 3.4: Kettle

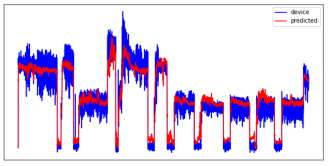


FIGURE 3.5: Refrigerator

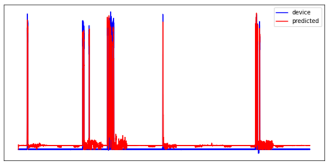


FIGURE 3.6: Microwave

Shown in Figures 3.4, 3.5 and 3.6 are some of the predictions for kettle, refrigerator and microwave respectively.

To validate the performance of our models we also tested the model using data from UK-DALE ([24]) and REDD ([25]) datasets. We tested the models in houses which are unobserved during training. The train/test split for different devices for the two datasets is shown in table 3.3. We also compare the performance of our models

TABLE 3.3: Houses used for training and testing

	UK-DALE dataset		REDD dataset	
	Training	Testing	Training	Testing
Refrigerator	1	2	2,6	1
Washing machine	1	2	3,6	1
Microwave	1	2	2,6	1
Kettle	1,3,5	2	-	-
Dishwasher	1,3	2	2,6	1

with the sequence to point architecture introduced in [19]. Our choice of [19] as the benchmark was motivated by the following reasons.

- An architecture with sequence-to-point output is expected to be more accurate compared to the same architecture with sequence-to-short sequence output. Hence, the performance gain from our approach can be established.

- Most of the recently proposed NILM solutions have used [19] for performance comparison.
- Results for both REDD and UK-DALE datasets, which we used as datasets for experiments, are presented in [19].

Accordingly, we compare the performance of the three models given below.

- Model (1): seq2point architecture in [19]
- Model (2): our approach with $s = 1008$
- Model (3): our approach with $s = 508$

Each model is used to infer the appliance level power consumption of a refrigerator (FRDG), washing machine (WASH), microwave (MWAV), kettle (KETL), and dishwasher (DWSR). Table 3.4 and Table 3.5 show the performance of the models for UK-DALE and REDD datasets respectively. Considering Table 3.4, it can be seen that model (3) outperforms the seq2point architecture proposed in [19] in terms of MAE for all the appliances. Model (1) outperforms model (3) in terms of SAE for refrigerator, washing machine and microwave. When the appliance signatures are lost in the noise caused by other appliances, model (1) tends to better predict the average power consumption of the appliance. Hence, the superior performance in terms of SAE can be observed. The SAE performance of our models can be improved by incorporating SAE to the training loss function.

The event detection capability of model (3) is superior to model (1) except in microwave as depicted by the F1 scores. The poor performance of all the models for microwave is due to the absence of a distinctive signature which can separate the microwave from other devices. When $s = 508$, the input window captures a significant amount of timestamps after and before each timestamp of the output window. In contrast, when $s = 1008$, only a few timestamps after a given timestamp of the output window are captured by the input window. Hence in model (2), the inferences are done without observing the complete signature of the appliance which can lead to confusions among appliances. On the other hand, in the considered setting when $s = 1008$, assuming the sampling and the processing delays are negligible, the output window will be generated 48 seconds after sampling the earliest of the corresponding inputs whereas the respective time when $s = 508$ is 25 minutes. Hence, in

a real-time setting, model (2) can be used for event detection and disaggregation in real-time where the inferences can be improved later using model (3). Since model (3) performed better than model (1) for UK-DALE data, we did not apply model (1) to REDD dataset (Table 3.5). A similar structure has been followed in [19].

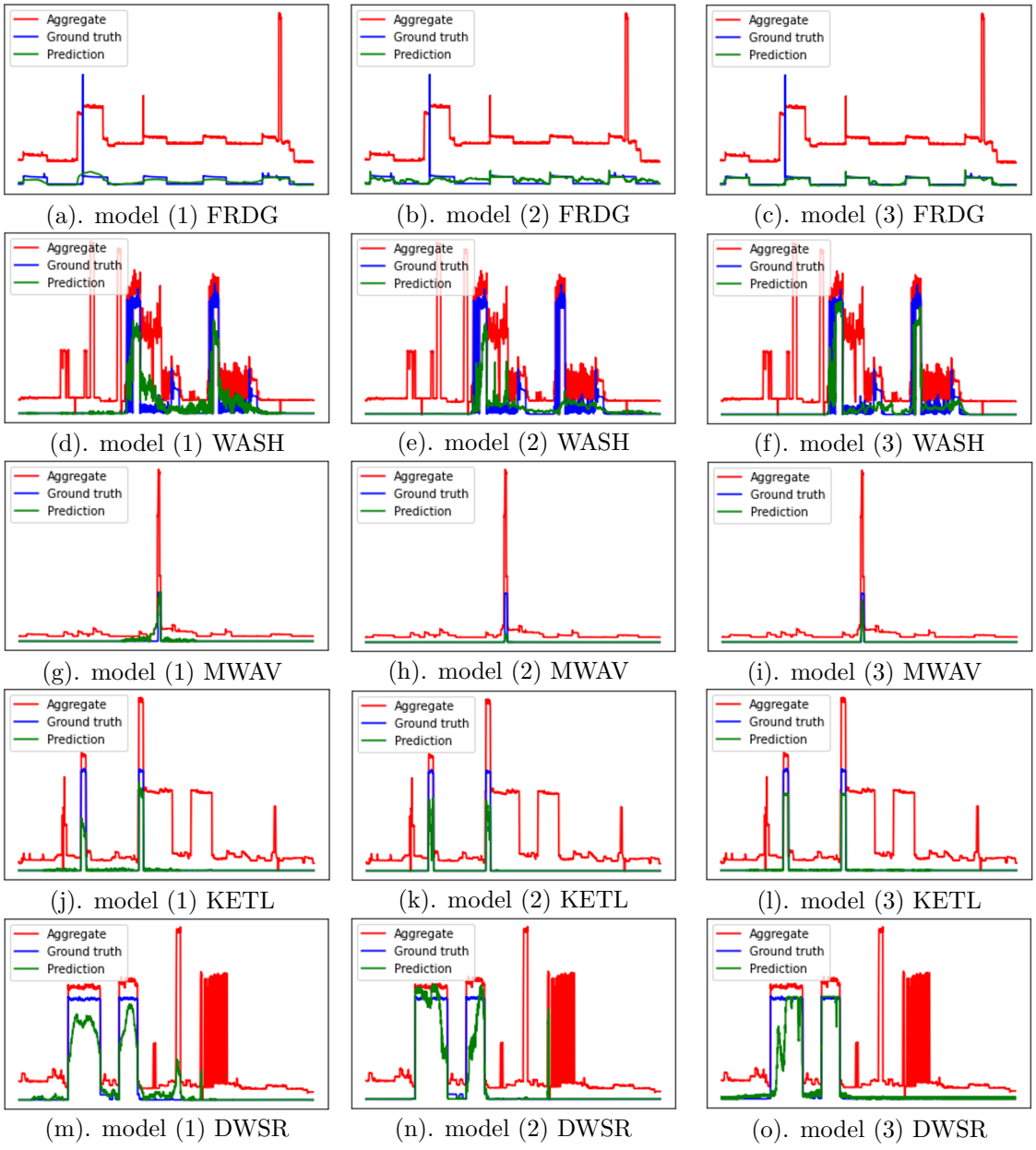
Figure 8 depicts example disaggregations on the UK-DALE data for the five devices using the three models. Each subfigure shows the aggregate power signal considered for disaggregation, the ground truth (ideal dissaggregation) and the predicted dissaggregation by our approach. The superior disaggregation performance of model (3) can be observed through the examples.

TABLE 3.4: The evaluation of the models for the UK-DALE dataset

Metric	Model	FRDG	WASH	MWAV	KETL	DWSR
MAE	(1)	24.45	10.77	11.24	20.55	24.20
	(2)	21.10	12.00	8.42	16.82	35.80
	(3)	**16.01**	**8.80**	**7.90**	**9.34**	**21.31**
SAE	(1)	**0.082**	**0.020**	**0.385**	0.354	0.063
	(2)	0.212	0.264	0.526	0.271	0.366
	(3)	0.240	0.250	0.563	**0.242**	**0.013**
Recall	(1)	**0.8590**	0.9246	**0.7060**	0.9678	**0.9140**
	(2)	0.7880	0.8868	0.1480	0.9780	0.4827
	(3)	0.8587	**0.9330**	0.1792	**0.9809**	0.9036
Precision	(1)	0.8299	0.3369	0.2990	0.9021	0.2941
	(2)	0.8119	0.3077	0.3881	0.9330	**0.7646**
	(3)	**0.9205**	**0.6211**	**0.4478**	**0.9602**	0.5843
Accuracy	(1)	0.8589	0.9780	0.9900	0.9986	0.9354
	(2)	0.8245	0.9757	0.9942	**1.0000**	**0.9811**
	(3)	**0.9041**	**0.9926**	**0.9947**	0.9994	0.9790
F1 score	(1)	0.8441	0.4939	**0.4201**	0.9338	0.4450
	(2)	0.7998	0.4568	0.2070	0.9550	0.5918
	(3)	**0.8885**	**0.7458**	0.2560	**0.9705**	**0.7097**

3.2. Calibration results

The results of the current transformer calibration procedure are shown below. The table 3.6 depicts the line fitting parameters described in Section 2.2.4. Different appliance combinations are used for each trial. The least square fit for V_{out} vs V_{cin}



(a). model (1) FRDG (b). model (2) FRDG (c). model (3) FRDG

(d). model (1) WASH (e). model (2) WASH (f). model (3) WASH

(g). model (1) MWAV (h). model (2) MWAV (i). model (3) MWAV

(j). model (1) KETL (k). model (2) KETL (l). model (3) KETL

(m). model (1) DWSR (n). model (2) DWSR (o). model (3) DWSR

FIGURE 3.7: The disaggregation results on UK-DALE dataset for refrigerator, washing machine, microwave, kettle and dish washer.

TABLE 3.5: The evaluation of the models for the REDD dataset

Metric	Model	FRDG	WASH	MWAV	DWSR
MAE	(2)	23.49	40.18	14.45	5.12
	(3)	**22.14**	**40.12**	**9.56**	**5.10**
SAE	(2)	0.054	**0.387**	**0.254**	**0.126**
	(3)	**0.042**	0.439	0.328	0.285
Recall	(2)	**0.9413**	0.5894	**0.9891**	**0.7508**
	(3)	0.9301	**0.5266**	0.7224	0.7169
Precision	(2)	0.9243	0.8114	0.3210	0.6085
	(3)	**0.9513**	**0.9848**	**0.8983**	**0.4808**
Accuracy	(2)	0.9393	0.9774	0.9918	**0.9911**
	(3)	**0.9476**	**0.9799**	**0.9985**	0.9872
F1 score	(2)	0.9327	0.6828	0.4847	**0.6722**
	(3)	**0.9406**	**0.6863**	**0.8009**	0.5756

is shown in figure 3.8. The least squares fit found using the above procedure is $V_{out} = 16.7V_{cin} + 0.28$.

Trial	**Rice cooker**	**Heater 1**	**Heater 2**	**Iron**	V_{out}**(mV)**	A_{in}**(A)**
1	ON	OFF	OFF	OFF	37.3	2.21
2	OFF	ON	OFF	OFF	78	4.66
3	OFF	OFF	OFF	ON	86.6	5.17
4	OFF	OFF	ON	OFF	101	6.05
5	ON	ON	OFF	OFF	113	6.8
6	ON	OFF	OFF	ON	121	7.27
7	ON	OFF	ON	OFF	135	8.1

TABLE 3.6: Line fitting parameters

3.3. Active power monitor specifications

As described in Section 2.2, the active power monitor is capable of sampling voltage and current with 1kHz frequency. The power samples can be obtained with (1/3)

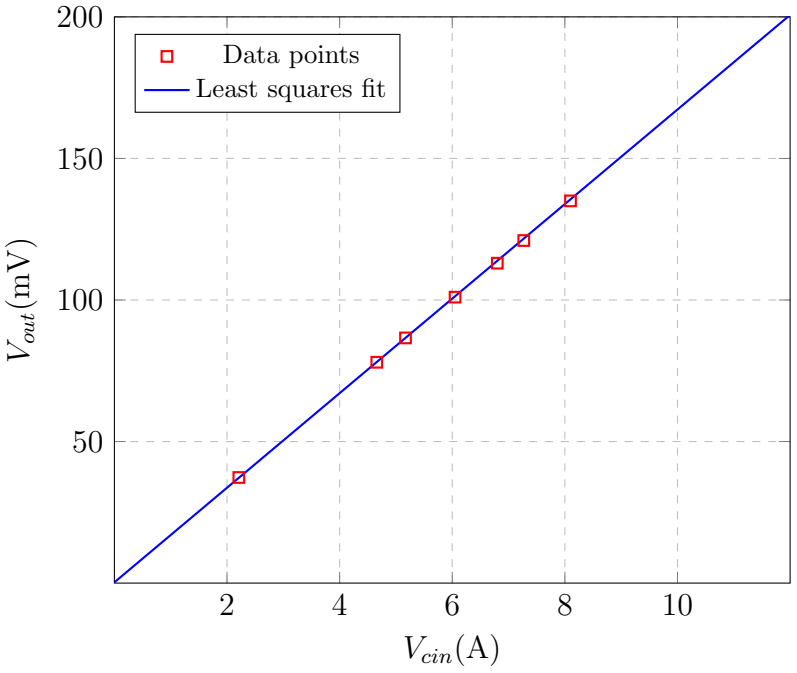


FIGURE 3.8: The least squares fit of V_{out} vs V_{cin}

Hz frequency. In addition, Table 3.7 shows the power measurement accuracy of the active power monitor. Here, the average power measured by the active power monitor is compared with the rated power consumption of the appliances. Although the average percentage deviation obtained through this experiment is around 4.8%, the actual deviation is lesser since the actual power consumption of appliances is lesser than the rated value. In fact, the theoretically calculated deviation using the maximum possible errors of the circuit components is around 3%. The cost of manufacturing the active power monitor is around Rs. 5000.

Appliance	**Rated value (W)**	**Measured value (W)**	**Deviation (%)**
Kettle	1000	900	10
Microwave	1150	1100	4.5
Fridge	100	100	0

TABLE 3.7: Power measurement accuracy

Chapter 4

DISCUSSION AND CONCLUSIONS

In this section we present a discussion about the principle results of the project along with the generalizations we can make based on what we encountered during practical implementations. First, we present the discussion for each component of the NILM system separately. Then we present the overall conclusion of the project.

4.1. Disaggregation algorithm

We have evaluated the disaggregation performance of our algorithm for three devices used in typical Sri Lankan households. In addition, we have evaluated the performance for two more devices using the data from datasets. More appliances such as water pump and air conditioner can be exploited as future work. Some of the major conclusions derived through the results are,

- From the results for standard datasets, it is clear that deep learning models generalize over different houses, different appliance combinations and target appliances with different rated values as exploited by Neural NILM[17].
- There is a trade off between accuracy and real-time performance. We have proposed to use two modelling variations together to enhance both factors although in this approach the real-time disaggregation performance can be less. Although we used only the active power for modelling due to the availability of datasets, reactive power can be used to improve both the real-time performance and the accuracy. Moreover, the high frequency signatures can greatly improve the performance. Although, it is infeasible to process the high frequency signatures in the cloud due to the high bandwidth requirement for transmission, they can be processed on device using a powerful micro-controller solution. The results of this processing can be used to improve the accuracy of the cloud algorithm.
- The high disaggregation performance for custom data suggests that fine tuning the models for each house can greatly improve the results. This can be

achieved by providing an additional smart meter to the users apart from the aggregate power monitor. The users can connect the smart meter to a particular device where the fine tuning can be automated through a cloud algorithm. In fact this is another advantage of having a separate model to infer the power consumption of each device. The models for separate devices can be fine tuned independently.

- The data synthesis technique can be used to increase both the quality and quantity of training data. This is vital in optimizing deep learning based architectures.

Additional use cases of our system are remote home electricity monitoring and tenant energy monitoring for landlords.

4.2. Electronic power monitor

Following are some key conclusions out of power monitor design.

- The Active power monitoring SoCs such as ICs from ADE series is an alternative to our custom circuit for active power sampling. But such ICs require the current to pass through the IC making the device intrusive.
- Although we designed the circuit to have a Real Time Clock module to retrieve timestamp, we later identified using the UTC timestamp that can be retrieved using the WiFi connection is more easier and robust.
- The Voltage Signal processing can be done without an Instrumentation amplifier by introducing a shift after step down. But the denoising effect of the amplifier improves the signal quality.

Some key problems we encountered and conclusions made during microcontroller programming are given below.

- The accuracy of active power samples can be increased by using high voltage and current sampling frequencies. Although we use 1 kHz sampling frequency, a sampling frequency around 10 kHz will yield more accurate and less noisy active power samples.

- Timer ISR routines in the ESP32 module cannot be pushed beyond 1kHz frequency. The ISR instances queue get overloaded in the core causing a watchdog timer trigger.
- The work division between cores reduces core overloading. Tasks of similar gravity such as timer ISR and WiFi transmission should be divided between the two cores to prevent watchdog timer triggers.
- ADC channel 0 of ESP32 module is unstable during WiFi operations. Therefore we used the ADC channel 1 with minor adjustments to the circuit.

4.3. Data pipeline and web application development

The conclusions we can draw from the cloud based data pipeline implementation along with the exceptions and problems can be summarized as follows.

- The cloud based processing for end to end products such as this is convenient from the customer's point of view as well as the product developer's point of view since,
  - Re-deployment of improved/updated deep learning models is easier
  - Computational power required for deep learning models is readily available
  - The electronic power monitor can be developed economically
  - Improved flexibility to the developer when optimizing the data pipeline
  - Improved system security
- The Firebase database solution enables rapid prototyping for a project like ours as it offers Backend as a service (BAAS).
- React can be identified as a popular front end library that enables fast web application development flexible to many requirements.

4.4. Overall conclusion

As described under the problem statement, the aim of our project is to deliver a system that helps the users to reduce the electricity consumption in their household using NILM. We have successfully designed and evaluated the four major deliverables of the project. The disaggregation algorithm outperforms the state of the art disaggregation algorithms when tested using standard NILM datasets. The active power monitor is capable of measuring power with an error of around 3%. We have demonstrated the end to end functionality of the system using real-time disaggregation for custom data collected using the power monitor and displaying the results using the web application. Under the discussion and conclusion section, we have discussed possible improvements for each component, which can be implemented as future extensions.

References

[1] Fatih Issi and Orhan Kaplan. The determination of load profiles and power consumptions of home appliances. *Energies*, 11(3), 2018. ISSN 1996-1073.

[2] Bp statistical review of world energy. 2016.

[3] Carrie Armel, Abhay Gupta, Gireesh Shrimali, and Adrian Albert. Is disaggregation the holy grail of energy efficiency? the case of electricity. *Energy Policy*, 52:213–234, 01 2013. doi: 10.1016/j.enpol.2012.08.062.

[4] A. Ridi, C. Gisler, and J. Hennebert. A survey on intrusive load monitoring for appliance recognition. In *2014 22nd International Conference on Pattern Recognition*, pages 3702–3707, 2014. doi: 10.1109/ICPR.2014.636.

[5] G. W. Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992. doi: 10.1109/5.192069.

[6] C. Duarte, P. Delmar, K. W. Goossen, K. Barner, and E. Gomez-Luna. Non-intrusive load monitoring based on switching voltage transients and wavelet transforms. In *2012 Future of Instrumentation International Workshop (FIIW) Proceedings*, pages 1–4, 2012. doi: 10.1109/FIIW.2012.6378333.

[7] C. Laughman, Kwangduk Lee, R. Cox, S. Shaw, S. Leeb, L. Norford, and P. Armstrong. Power signature analysis. *IEEE Power and Energy Magazine*, 1 (2):56–63, 2003. doi: 10.1109/MPAE.2003.1192027.

[8] Shwetak Patel, Thomas Robertson, Julie Kientz, Matthew Reynolds, and Gregory Abowd. At the flick of a switch: detecting and classifying unique electrical events on the residential power line. pages 271–288, 09 2007.

[9] H. Y. Lam, G. S. K. Fung, and W. K. Lee. A novel method to construct taxonomy electrical appliances based on load signaturesof. *IEEE Transactions on Consumer Electronics*, 53(2):653–660, 2007. doi: 10.1109/TCE.2007.381742.

[10] M. Z. A. Bhotto, S. Makonin, and I. V. Bajić. Load disaggregation based on aided linear integer programming. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 64(7):792–796, 2017. doi: 10.1109/TCSII.2016.2603479.

[11] M. Baranski and J. Voss. Genetic algorithm for pattern detection in nialm systems. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, volume 4, pages 3462–3468 vol.4, 2004. doi: 10.1109/ICSMC.2004.1400878.

[12] K. Srinivasarengan, Y. G. Goutam, M. G. Chandra, and S. Kadhe. A framework for non intrusive load monitoring using bayesian inference. In *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 427–432, 2013. doi: 10.1109/IMIS.2013.78.

[13] Huijuan Shao, Manish Marwah, and Naren Ramakrishnan. A temporal motif mining approach to unsupervised energy disaggregation: Applications to residential and commercial buildings. AAAI'13, page 1327–1333. AAAI Press, 2013.

[14] Hyungsul Kim, Manish Marwah, Martin Arlitt, Geoff Lyon, and Jiawei Han. Unsupervised disaggregation of low frequency power measurements. volume 11, pages 747–758, 04 2011. doi: 10.1137/1.9781611972818.64.

[15] R. Jia, Y. Gao, and C. J. Spanos. A fully unsupervised non-intrusive load monitoring framework. In *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 872–878, 2015. doi: 10.1109/SmartGridComm.2015.7436411.

[16] J. Zico Kolter, Siddarth Batra, and Andrew Y. Ng. Energy disaggregation via discriminative sparse coding. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1*, NIPS'10, page 1153–1161, Red Hook, NY, USA, 2010. Curran Associates Inc.

[17] Jack Kelly and William Knottenbelt. Neural nilm: Deep neural networks applied to energy disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, BuildSys '15, page 55–64, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450339810. doi: 10.1145/2821650.2821672. URL `https://doi.org/10.1145/2821650.2821672`.

[18] G. Zhou, Z. Li, M. Fu, Y. Feng, X. Wang, and C. Huang. Sequence-to-sequence load disaggregation using multiscale residual neural network. *IEEE Transactions on Instrumentation and Measurement*, 70:1–10, 2021. doi: 10.1109/TIM.2020.3034989.

[19] Chaoyun Zhang, Mingjun Zhong, Zongzuo Wang, Nigel Goddard, and Charles Sutton. Sequence-to-point learning with neural networks for non-intrusive load monitoring. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. URL `https://ojs.aaai.org/index.php/AAAI/article/view/11873`.

[20] Ashish Vaswani et al. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, Dec. 2017. Curran Associates Inc. ISBN 9781510860964.

[21] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. *ArXiv*, abs/1508.04025, 2015.

[22] S. Welikala, C. Dinesh, M. P. B. Ekanayake, R. I. Godaliyadda, and J. Ekanayake. Incorporating appliance usage patterns for non-intrusive load monitoring and load forecasting. *IEEE Transactions on Smart Grid*, 10(1): 448–461, 2019. doi: 10.1109/TSG.2017.2743760.

[23] C. Dinesh, S. Makonin, and I. V. Bajic. Incorporating time-of-day usage patterns into non-intrusive load monitoring. In *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1110–1114, Nov. 2017.

[24] Jack Kelly and William Knottenbelt. The uk-dale dataset, domestic appliance-level electricity demand and whole-house demand from five uk homes. *Scientific Data*, 2, 03 2015. doi: 10.1038/sdata.2015.7.

[25] J. Zico Kolter and Matthew J. Johnson. Redd: A public data set for energy disaggregation research. In *IN SUSTKDD*, 2011.