Sequence-to-short Sequence Learning with Attention-based Autoencoders for Non-Intrusive Load Monitoring

Mevan Wijewardena[a], Shalitha Pathiranage[a], Thanuji Nanayakkara[a], Isuru Rajapakshe[a], Subodha Charles[a], Tharaka Samarasinghe[a,b]

[a]*Department of Electronic and Telecommunication Engineering, University of Moratuwa, Moratuwa 10400, Sri Lanka*
[b]*Department of Electrical and Electronic Engineering, The University of Melbourne, Parkville, VIC 3010, Australia*

Abstract

Non-intrusive load monitoring (NILM) involves inferring the appliance level power usage of a consumer given the aggregate power consumption. Different methodologies have been proposed for NILM and recently the focus has shifted towards deep learning-based approaches. These models utilize sliding window-based approaches for sequence modelling due to limitations in dealing with long input sequences. This paper proposes sequence-to-short sequence learning, where a sub-sequence of an appliance level power window corresponding to the input window is expected as the output from the learning model. Due to the recent success of attention-based models in sequence modelling, a novel attention-based autoencoder architecture with multiple attention heads is proposed. The attention heads can extract the relevant signatures of the input sequence by learning to forget the irrelevant signatures. The paper explores how the attention-based autoencoder architecture in conjunction with sequence-to-short sequence learning can be used to improve both the accuracy and the real-time capability of NILM systems. The performance of our model is evaluated by comparing it with a state-of-the-art model. In addition, the robustness of our model to noisy data is empirically evaluated using a custom dataset.

Keywords: Non-intrusive load monitoring, energy disaggregation, sequence-to-short sequence learning, attention-based autoencoders, multi-head attention.

1. Introduction

Load monitoring has been an extensively researched topic in smart energy systems [1, 2]. The power consumption data of individual appliances that can be obtained thanks to load monitoring techniques can be used to improve the energy efficiency of a system in various ways. Specifically, the high power consuming appliances can be identified, and they can be operated in a timely manner. Moreover, devices with anomalous power consumption can be identified [3] by analyzing the power consumption curves, and can then be repaired or discarded. To this end, monitoring can be done intrusively [2] or non-intrusively [1], where in the former case, the appliances are monitored using dedicated smart meters, and in the latter case, the appliance level power consumption is inferred from the aggregate power consumption. Although intrusive load monitoring (ILM) is an ideal solution for consumer feedback, the extensive hardware requirements impose constraints on its deployment feasibility. In contrast, non-intrusive load monitoring (NILM), which is also referred to as energy disaggregation, has considerably less stringent hardware requirements. However, the use of efficient and powerful algorithms is mandatory for accurate performance.

The NILM concept involves in finding the power consumption of each appliance without directly measuring it, through inference from the aggregate power consumption. NILM solutions in existing literature typically include a smart electricity meter that samples the aggregate power, and does the inference through statistical and machine learning models.

Related Work: Various techniques have evolved over the years for energy disaggregation [4, 5, 6, 7, 8]. Supervised learning approaches such as integer programming [4], genetic algorithms [5], knapsack algorithm-based approaches, Bayesian inference-based approaches [6], as well as unsupervised and semi supervised learning methods such as motif mining [9], factorial hidden Markov models [7, 10], unsupervised Bayesian learning [11], and discriminative sparse coding [12], have been explored in existing literature.

The latest research on NILM exploits deep learning-based approaches, thanks to the availability of high computational power and high volumes of data [8, 13, 14]. The work in this area was strengthened by Kelly et al. [8], who established the superiority of deep learning-based methods for energy disaggregation compared to classical methods such as hidden Markov models.

In deep learning-based methods, a dedicated model is trained to infer the power consumption of each appliance through the aggregate power consumption [8, 13, 14]. Moreover, due to the vanishing and exploding gradient problems

that arise when dealing with long input sequences, sliding window-based methods are used in each model. The input to the model is generated by sliding a window through the aggregate power sequence from which a window of the target appliance level power sequence is expected as the output.

The sliding window methods vary depending on the relative lengths of the input and output windows. In *sequence-to-sequence learning*, the windows have the same length and the output window corresponds in time to the input window [15]. In *sequence-to-point learning*, a single point of the target appliance level power sequence is expected as the output [13]. Typically this point corresponds in time to the midpoint of the input window. Sequence-to-point learning has high computational complexity during inference although it is superior in terms of accuracy. In sequence-to-sequence learning, the accuracy of the inferred values towards the beginning and the end of the output window are low. Hence, we adopt *sequence-to-short sequence learning* where a sub-sequence of the appliance level power window corresponding to the input window is expected as the output. This method has lower computational complexity compared to sequence-to-point learning and higher accuracy compared to sequence-to-sequence learning. The work of Liang et al. [16] uses sequence-to-short sequence learning to infer the output window that has half the length of the input window. In our implementation, we infer output windows that are smaller in length compared to the input window, which leads to an accuracy comparable to sequence-to-point learning. In addition to optimizing in terms of accuracy, this approach can be used to improve the real-time performance of the system as well. Hence, we introduce two variations of sequence-to-short sequence learning where the first variation has superior real-time capability and the second variation has superior accuracy.

Primarily, long short term memory (LSTM) and gated recurrent unit (GRU) networks are used in the literature to infer the appliance level power window given the aggregate power window. One dimensional convolutional neural networks (1DCNN) are also used due to their feature extraction capabilities [17]. Inspired by the recent success of attention-based architectures in sequence modelling tasks such as neural machine translation [18], in this paper, we propose a novel attention-based autoencoder model that combines the feature extraction capabilities of 1DCNN, and sequence modelling capabilities of the attention mechanism and LSTM. A model with a single attention layer is considered in [19] for energy disaggregation. Our work is different, since we exploit the use of multi-head attention introduced in [18]. Implementing multiple attention heads is beneficial in energy disaggregation as it involves filtering out the power signatures of the target appliance from a pool of power signatures resulting from multiple

appliances.

In order to ensure the suitability for different settings, we train and validate the proposed model on two state-of-the-art NILM datasets collected in two regions. Moreover, we empirically evaluate the robustness of our models to noisy data using a custom dataset.

Contributions:

- We propose a novel attention-based autoencoder model with multiple attention heads for NILM.
- We effectively combine the attention-based autoencoder model with sequence-to-short sequence learning to improve both accuracy and real-time capability of NILM.
- We effectively combine data pre-processing with a novel data synthesis technique during training.
- We empirically validate the superior performance of the model compared to the state-of-the-art energy disaggregation algorithms.

Structure: The remainder of the paper is organized as follows. First we define the NILM problem and present a high level overview of our approach in Section 2. Then we introduce our model architecture in Section 3 followed by the model training and inference process in Section 4. We introduce the experimental setup and present the numerical results in Section 5. Finally, Section 6 concludes the paper.

Notations: Following notations are used throught the paper. The (i, j)-th element and i-th row of a matrix $\mathbf{\Theta}$ is denoted by $\Theta_{(i,j)}$, $\mathbf{\Theta}_{(i)}$, respectively. Similarly, (i, j)-th element and i-th row of a matrix $\mathbf{\Theta}_k$ is denoted by $\Theta_{k(i,j)}$, $\mathbf{\Theta}_{k(i)}$, respectively. Consider a sequence $\tilde{S}$. A sub-sequence of $\tilde{S}$ of length l starting from index j is denoted by $\tilde{S}[j : j + l]$.

2. Problem Formulation

Consider a typical household that consists of a multitude of multi-state appliances such as washing machines, refrigerators and kettles. Figure 1 illustrates the energy disaggregation process for such a household, which can be formulated as follows. 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 $p_i(t)$, for

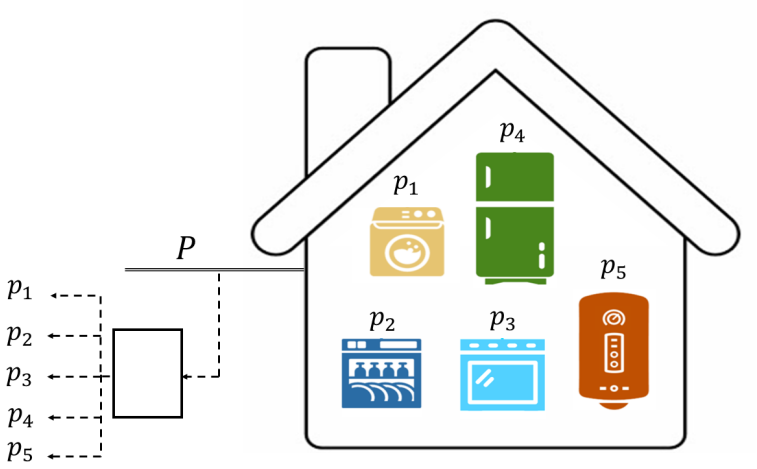


Figure 1: Non-intrusive load monitoring concept. The goal is to infer the power consumption by each appliance (p_i), without directly measuring each p_i, using only the aggregate power consumption data (P).

$1 \leqslant i \leqslant N$, 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

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

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 $j \in \mathbb{Z}$, which we refer to as timestamps, 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 $j \in \mathbb{Z}$ and $i \in [1, N]$. We note that the majority of household appliances can be categorized as multi-state appliances. We omit appliances with continuously varying power demands such as lighting and laptop/phone chargers in our modeling due to their "on time" being long and unpredictable.

We adopt a deep learning-based model for energy disaggregation. It is well known that deep neural networks suffer from vanishing and exploding gradient problems when the inputs to the model are long sequences. Hence, we resort to a sliding window-based approach for feeding data to the model. To this end, the input to the deep learning-based model of each appliance is generated by sliding a window through the aggregate power sequence, and a sub-sequence of the corresponding appliance level power window is expected as the output. This approach,

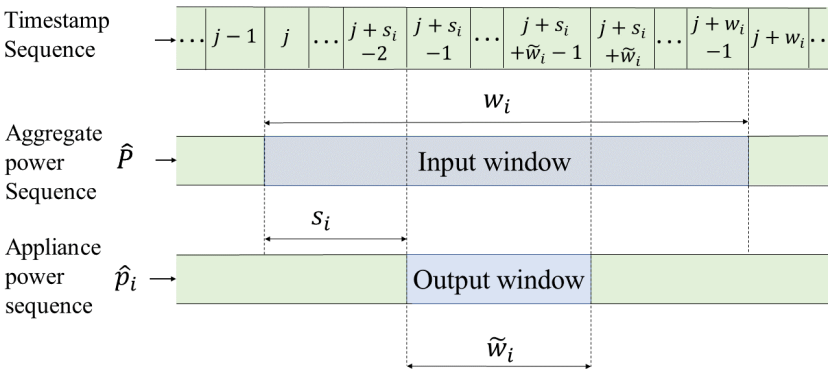


Figure 2: The sequence-to-short sequence learning concept.

which is known as sequence-to-short sequence learning, is illustrated in Figure 2. The input to the model for the i-th, $1 \leqslant i \leqslant N$, appliance is $\hat{P}[j : j + w_i]$, where $j \in [1, l - w_i + 1]$, l is the length of the power sequence and w_i is the length of the input window of the i-th appliance. The input $\hat{P}[j : j + w_i]$ is used to infer the expected output from the model which is $\hat{p}_i[j + s_i : j + s_i + \tilde{w}_i]$, where $\tilde{w}_i \leqslant w_i$ is the length of the output window of the i-th appliance and $s_i \in [0, w_i - \tilde{w}_i]$ is illustrated in Figure 2. Setting $\tilde{w}_i$ to a small value results in either high accuracy or high real-time capability depending on the value of s_i, as described below.

From Figure 2 observe that, the output for the $(j + s_i - 1)$-th timestamp is generated after sampling the input for the $(j + w_i - 1)$-th timestamp. Hence, for the $(j + s_i - 1)$-th timestamp, the output is generated $(w_i - s_i)T + T_{\text{proc}}$ after sampling the input, where T_{proc} is the time taken for inference and other processing tasks. Hence, when $\tilde{w}_i$ is small, increasing s_i is favourable for real-time performance. Moreover, when $\tilde{w}_i$ is small and s_i is close to $w_i/2$, the input window captures a significant amount of 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 variation 1 (V1) we use s_i close to w_i and in variation 2 (V2) we use s_i close to $w_i/2$. Both of the above variations are implemented using the attention-based autoencoder architecture described in Section 3.

3. Model Architecture

This section outlines our architecture which consists of the output soft-max scaling layers and the autoencoder with attention layers. The architecture overview is shown in Figure 3.

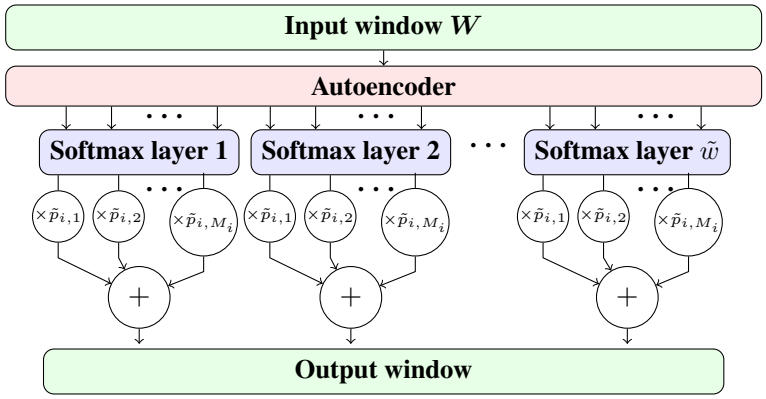


Figure 3: The attention-based autoencoder architecture for the i-th appliance.

3.1. Soft-max scaling layers

Let the number of states of the i-th appliance be M_i and the mean power consumption of the u-th $(1 \leqslant u \leqslant M_i)$ state be $\tilde{p}_{i,u}$. Let $\boldsymbol{O}_i$ and $\boldsymbol{W}_i$ be the random vectors denoting the input and output windows. Then the expected value of $\boldsymbol{O}_i$ given $\boldsymbol{W}_i$ can be written as,

$$\mathbb{E}\{\boldsymbol{O}_i|\boldsymbol{W}_i\} = \boldsymbol{F}_i\tilde{\boldsymbol{p}}_i, \tag{2}$$

where $\boldsymbol{F}_i \in \mathbb{R}^{\tilde{w}_i \times M}$ is a matrix whose element (t, u) is $\Pr(\Phi_{i,t} = u|\boldsymbol{W}_i)$, $\Phi_{i,t}$ is the random variable representing the state of the appliance at the t-th timestamp of the output window and $\tilde{\boldsymbol{p}}_i \in \mathbb{R}^{M_i \times 1}$ is a vector whose u-th element is $\tilde{p}_{i,u}$. To model $\boldsymbol{F}_i$, $\tilde{w}_i$ soft-max scaling layers are added at the model output as shown in Figure 3, where Softmax layer $t(1 \leqslant t \leqslant \tilde{w}_i)$ outputs the t-th row of matrix $\boldsymbol{F}_i$. The expected output window can then be calculated from the softmax layer outputs using Equation (2).

3.2. Autoencoder

As shown in Figure 4, the autoencoder consists of an encoder with convolutional layers, and a decoder with convolutional, multi-head attention [18] and LSTM layers. The convolutional encoder extracts the signatures of the appliances and presents them to the decoder. We propose multi-head attention layers for the decoder which are capable of filtering out the signatures of the target appliance from a pool of signatures resulting from multiple appliances.

The outputs from all the convolutional layers are zero padded in order to ensure equal dimensions for the input and the output of the layer. Moreover, the

LSTM layers are set to output the predicted state vector at all time-steps. A multi-head attention layer consist of several attention layers , both of which are described below.

Multi-head attention layer: The architecture of the multi-head attention layer is shown in Figure 5 [18]. The inputs to a multi-head attention layer can be given by the two matrices $\boldsymbol{Q}_i$ and $\boldsymbol{V}_i$, each having dimensions of $\tilde{w}_i \times \tilde{f}_i$. The input matrices are sent through independent feed-forward layers to obtain H_i pairs of $\tilde{w}_i \times \tilde{f}_i$ matrices ($\tilde{\boldsymbol{Q}}_{i,m}$, $\tilde{\boldsymbol{V}}_{i,m}$), for $1 \leqslant m \leqslant H_i$ as shown in Figure 5. Each of the pairs act as an input to the independent attention layers described below, from which the obtained outputs are processed, as shown in Figure 5, to obtain the layer output $\boldsymbol{O}_i$ having the same shape as the inputs.

Attention layer: The architecture of an attention layer is shown in Figure 6. An attention layer takes two matrices $\tilde{\boldsymbol{Q}}_i$ and $\tilde{\boldsymbol{V}}_i$ of shape $\tilde{w}_i \times \tilde{f}_i$ as input, and maps them to an output matrix $\tilde{\boldsymbol{O}}_i$ having the same shape as the inputs. The rows of $\tilde{\boldsymbol{Q}}_i$ and $\tilde{\boldsymbol{V}}_i$ are named as queries and values, respectively. Both queries and values represent learnt appliance signatures at different timestamps. The first step of attention is calculating the similarity score matrix $\boldsymbol{\Psi}_i$ between $\tilde{\boldsymbol{Q}}_i$ and $\tilde{\boldsymbol{V}}_i$, where $\Psi_{i(n,j)} = \boldsymbol{d}_{i,1}^{\top} \tanh(\boldsymbol{\Phi}_{i,1}\tilde{\boldsymbol{Q}}_{i(n)}^{\top} + \boldsymbol{\Phi}_{i,2}\tilde{\boldsymbol{V}}_{i(j)}^{\top} + \boldsymbol{d}_{i,2})$ [20]. Here $\boldsymbol{\Phi}_{i,1} \in \mathbb{R}^{\tilde{g}_i \times \tilde{f}_i}$, $\boldsymbol{\Phi}_{i,2} \in \mathbb{R}^{\tilde{g}_i \times \tilde{f}_i}$, $\boldsymbol{d}_{i,1} \in \mathbb{R}^{\tilde{g}_i \times 1}$ and $\boldsymbol{d}_{i,2} \in \mathbb{R}^{\tilde{g}_i \times 1}$ are learnt during training, such that $\Psi_{i(n,j)}$ represents a score of similarity between the two vectors $\tilde{\boldsymbol{Q}}_{i(n)}$ and $\tilde{\boldsymbol{V}}_{i(j)}$. Secondly, we obtain the normalized similarity matrix $\boldsymbol{\Omega}_i$ where,

$$\Omega_{i(n,j)} = \frac{\exp(\Psi_{i(n,j)})}{\sum_{p=1}^{\tilde{w}_i} \exp(\Psi_{i(n,p)})}. \tag{3}$$

Finally, the the output $\tilde{\boldsymbol{O}}_i$ is computed using $\tilde{\boldsymbol{O}}_i = \boldsymbol{\Omega}_i \tilde{\boldsymbol{V}}_i$. Observe that, the attention layer outputs a matrix whose t-th row is computed as a weighted sum of the values, where the weight assigned to each value depends on it's similarity to the t-th query. Hence, the parameters of the attention layer can be learnt such that, each row of the output matrix represents a weighted sum of the signatures, where larger weights are assigned to relevant signatures.

4. Model Training and Inference

This section gives an overview of the training and inference process that is used with the model outlined in Section 3 and elaborates the data synthesis technique.

4.1. Overview

Figure 7 shows an overview of the model training methodology. Data required for model training can be obtained either by implementing a custom data collection framework or by using already available NILM datasets. Typically, NILM datasets consist of simultaneous readings of the aggregate power signal and readings of the power signals of individual appliances of several households paired with a timestamp [21, 22]. We train the models to predict the output window di-

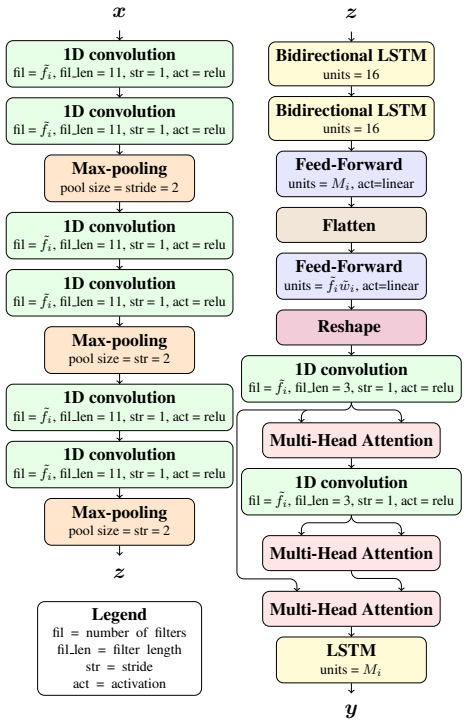


Figure 4: Autoencoder.

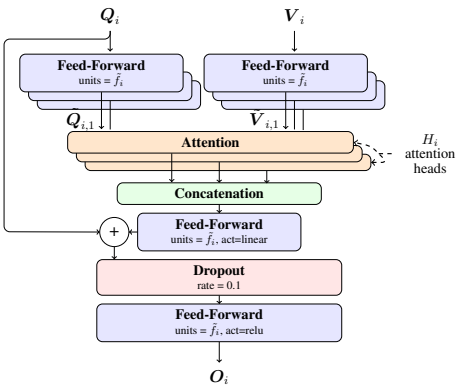


Figure 5: Multi-Head Attention.

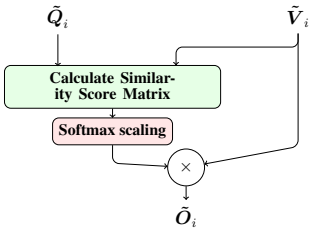


Figure 6: Attention Layer.

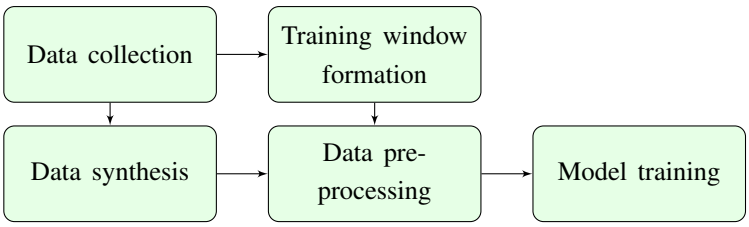


Figure 7: The overview of the proposed model training methodology.

rectly, where $\tilde{\boldsymbol{p}}_i$ is implemented using a feed-forward layer which is learnt during

training. We use mean squared error as the loss function. Alternatively, the models can be trained to predict $\boldsymbol{F}_i$, where the state distribution for the appliance level power values in the dataset and $\tilde{\boldsymbol{p}}_i$ are obtained using Gaussian mixture models. In this case, categorical cross entropy is used as the loss function. Both approaches lead to similar performance.

The aggregate and target appliance level power window pairs required for model training are formed from the datasets. In addition, training window pairs are synthesized (Section 4.2) in order to improve both the quality and the quantity of data. The aggregate and target appliance level windows are then scaled linearly such that they lie in $[0, 1]$. Formally, the scaled window for $\hat{p}_i[j : j + \hat{w}_i]$ is given by $\hat{p}_{i,\text{scaled}}[j : j + \hat{w}_i]$, where $\hat{p}_{i,\text{scaled}}[j + t] = (\hat{p}_i[j + t] - p_i^{\text{min}})/(p_i^{\text{max}} - p_i^{\text{min}})$ for $0 \leqslant t < \tilde{w}$ and p_i^{max} and p_i^{min} represent the maximum and minimum power values of $\hat{p}_i$, respectively. The aggregate power window is scaled similarly. The scaling will reduce vanishing and exploding gradient problems during training.

The attention-based autoencoder model described in Section 3 is trained on a mixture of real and synthetic data. The data windows are fed in mini-batches during training. During inference, the inputs to the model are scaled similar to the training phase using the scaling parameters obtained during training. After inference, the predicted appliance level power window $y_{i,\text{rescaled}}[j : j + \tilde{w}]$ is obtained by letting, $y_{i,\text{rescaled}}[j + t] = (p_i^{\text{max}} - p_i^{\text{min}})y_i[j + t] + p_i^{\text{min}}$ for $0 \leqslant t < \tilde{w}$, where $y_i[j : j + \tilde{w}]$ represents the window output from the model.

4.2. Data Synthesis

We adopt a data synthesis technique where training data is synthesized from the available data described in Section 4.1. 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. 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 ($1 \leqslant i \leqslant N$) appliance from the same house is denoted by Y. During data synthesis, X and Y are populated by synthesized aggregate power windows and the corresponding appliance level power windows of the i-th ($1 \leqslant i \leqslant N$) appliance. The data synthesis for the other appliances and houses follow the same procedure.

The steps taken for training data synthesis for the i-th appliance in a given house is outlined Algorithm 1. The appliance level power windows are generated by sliding a window of length w_i through $\hat{p}_i$ with a shift $\tilde{w}_i$ 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

$$\boldsymbol{x} = \boldsymbol{y} + \sum_{\substack{m=1 \\ m \neq i}}^{N} \hat{p}_k[b_m : b_m + w_i], \tag{4}$$

where b_m is chosen uniformly such that, $1 \leqslant b_m \leqslant l - w_i + 1$. Then, we add $\boldsymbol{x}$ and $\boldsymbol{y}[s_i : s_i + \tilde{w}_i]$ to X and Y respectively for each $\boldsymbol{x}$ (line 15-22), where s_i is defined in Section 2. The selective discarding of appliance level power windows prevents the model from learning to predict flat power signals arising from the skewed data distribution of some appliances due to 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. Time dependencies between appliances can be exploited to improve the performance of appliance combinations such as toaster and kettle which are typically operated within a short time gap [23, 24]. But the performance of appliances which do not exhibit such dependencies will be reduced, which will reduce the model generalizability.

5. Experiments

In this section, we describe the experiments conducted to evaluate the performance of our method. We evaluate the two variations of sequence-to-short sequence learning described in Section 2. We compare V2 with a state-of-the-art model in terms of accuracy to establish the superiority of our model. Then we highlight the utility of V1 for a real-time setting while discussing the trade-off in terms of performance compared to V2. Finally, we evaluate the applicability of our models to custom data. Using the custom data, we also empirically evaluate the robustness of our models to noise.

5.1. Experimental Setup

We use REDD [22] and UK-DALE [21] datasets for training and evaluation purposes. UK-DALE and REDD datasets have appliance level power readings sampled at a period of 6 and 3 seconds, respectively. In order to eliminate the inconsistencies of sampling rate in the dataset, we first upsample the data to 1

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

Data: $\hat{p}_m$ for $1 \leqslant m \leqslant N$, 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 $0 \leqslant n < (l - w_i)/\tilde{w}_i$ **do**
  Set $f = (\tilde{w}_i n)$;
  for $0 \leqslant j < c$ **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 $1 \leqslant m \leqslant N$ and $m \neq i$ **do**
        Sample $b_m \in [1, l - w_i + 1]$;
        Add $\hat{p}_m[b_m : b_m + w_i]$ to $\boldsymbol{x}$;
      end
    end
    Add $\boldsymbol{x}$, $\boldsymbol{y}[s_i : s_i + \tilde{w}_i]$ to X,Y;
  end
end

second followed by downsampling to 3 seconds. During upsampling, a gap greater than 2 minutes between two consecutive timestamps in the original data is treated as missing data. Data from different houses are used in the training and testing stages as tabulated in Table 1. We consider the appliances refrigerator (FRDG), washing machine (WASH), microwave (MWAV), kettle (KETL), and dishwasher (DWSR) for the experiments.

Table 1: Houses used for training/testing and the parameters for different devices

	UKDALE House No.		REDD House No.		Parameters			
	Training	Testing	Training	Testing	M_i	p_i^{on}	P_i^{num}	P_i^{choose}
FRDG	1	2	2,6	1	4	50	10	1
WASH	1	2	3,6	1	4	50	20	0.5
MWAV	1	2	2,6	1	3	200	10	0.6
KETL	1,3,5	2	-	-	4	300	10	0.6
DWSR	1,3	2	2,6	1	4	50	10	0.5

Values for M_i, appliance switch on power threshold p_i^{on}, P_i^{choose} and P_i^{num} are given in Table 1. The parameter p_i^{on} is set by considering the average on/off power values of the i-th appliance. P_i^{thresh} is set equal to p_i^{on}. The parameters M_i, P_i^{num} and P_i^{choose} are set by experimenting with several values. An initial value for M_i can be set by considering the number of distinct states of the i-th appliance. Similarly, the average duration of a single operational cycle and the average on-time of the i-th appliance can be used to initialize P_i^{num} and P_i^{choose}, respectively. The training mini-batch size is set to 512 for all appliances. We consider appliance-independent values for the parameters $\tilde{f}_i$, $\tilde{g}_i$, H_i, w_i, $\tilde{w}_i$ and s_i, thus we drop the index i for brevity, and we set $\tilde{f}$, $\tilde{g}$, H, w and $\tilde{w}$ to 32, 4, 8, 1024 and 8, respectively. The parameters $\tilde{f}$ and H are set to be larger than the total number of distinct appliance signatures of all the appliances and the number of appliance signatures of a single appliance, respectively. On the other hand, setting these parameters too large may lead to overfitting. The parameter w is set such that the input window captures the complete duration of a single operational cycle of any appliance. We implement the two variations of sequence-to-short sequence learning by setting $s = 1008$ for V1 and $s = 508$ for V2. We also compare the performance of the two variations with the sequence-to-point architecture introduced in [13]. Our choice of [13] 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.

- The approach in [13] has been used as a performance benchmark in existing literature.
- Results for both REDD and UK-DALE datasets, which were used in our experiments, are presented in [13].

5.2. Experimental Results

As the main evaluation metrics, we use the disaggregation performance metrics mean absolute error

$$\text{MAE} = \frac{1}{L}\sum_{m=1}^{L}|\tilde{y}_m - y_m|, \tag{5}$$

and signal aggregate error

$$\text{SAE} = \frac{|\sum_{m=1}^{L}\tilde{y}_m - \sum_{m=1}^{L}y_m|}{\sum_{m=1}^{L}y_m}, \tag{6}$$

where y_m, $\tilde{y}_m$ and L denote the actual appliance power consumption at time m, the predicted appliance power consumption at time m and the length of the power sequences, respectively [13]. 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 an appliance over a period of time. Although MAE is a more intuitive metric in a NILM setting, SAE will be useful from a user point-of view as the 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_i^{on}. The above mentioned metrics are used to evaluate the following models.

- (M1): Sequence-to-point (seq2point) method in [13]
- (M2): Our approach - V1
- (M3): Our approach - V2

Table 2a and Table 2b tabulate the performance of the models for UK-DALE and REDD datasets, respectively. Considering Table 2a, it can be seen that (M3) outperforms the seq2point architecture proposed in [13] in terms of MAE for all

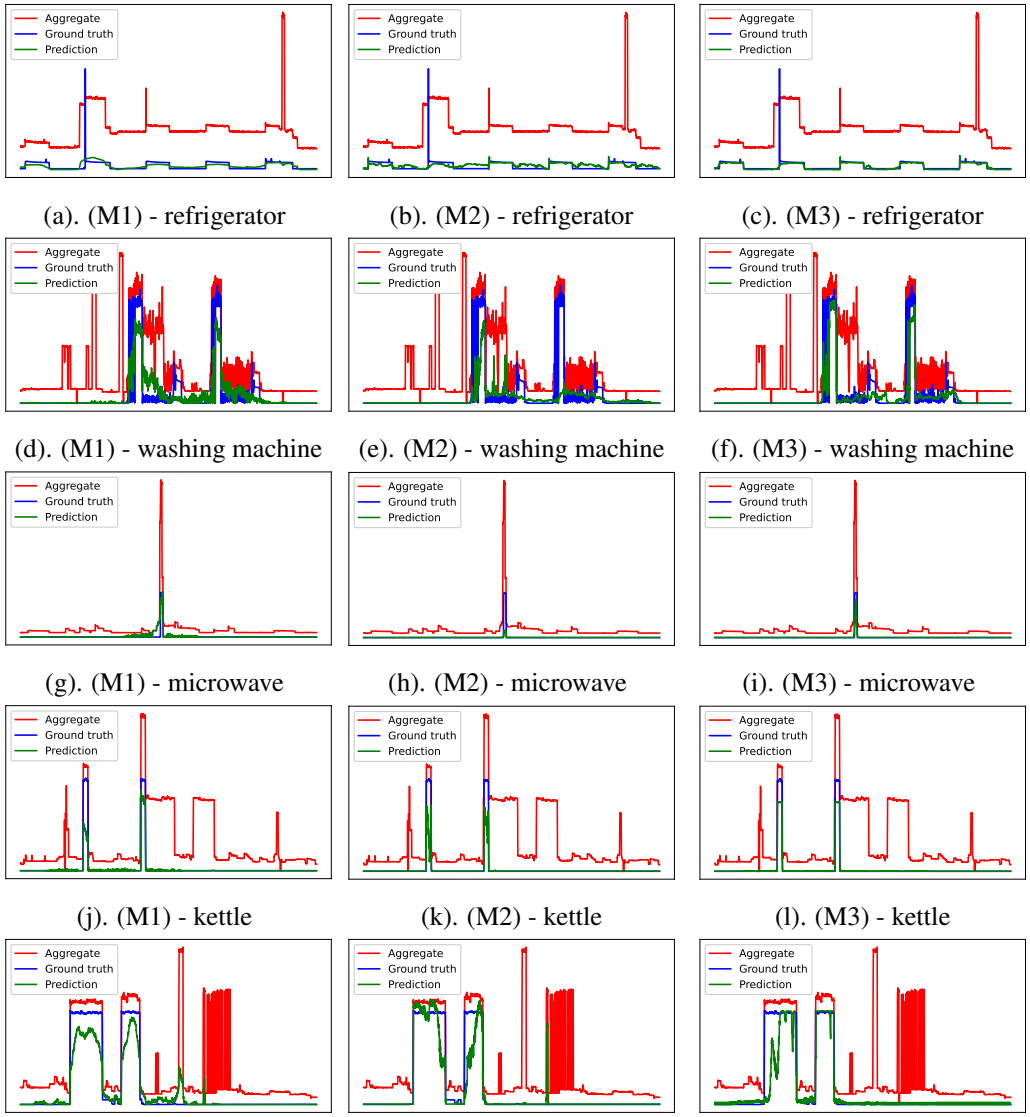
the appliances. (M1) outperforms (M3) in terms of SAE for refrigerator, washing machine and microwave. When the appliance signatures are lost in the noise caused by other appliances, (M1) tends to better predict the average power consumption of the appliance. Hence, the superior performance in terms of SAE can be observed. The SAE performance of our models can be improved further by incorporating SAE to the training loss function.

The event detection capability of (M3) is superior to (M1) except in the microwave scenario as depicted by the F1 scores. The poor performance of all the models for the microwave is due to the absence of a distinctive signature that 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 (M2), the inferences are done without observing the complete signature of the appliance, which can lead to confusions among appliances. On the other hand, when $s = 1008$, assuming the sampling and the processing delays are negligible, the output window is 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, (M2) can be used for event detection and disaggregation in near real-time where the inferences can be improved later using (M3). Since (M3) performed better than (M1) for UK-DALE data, we did not apply (M1) to REDD dataset (Table 2b). A similar structure has been followed in [13].

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 (M3) can be observed through the examples.

5.3. Evaluation On Custom Data

We also tested our models on custom data collected using the power monitoring hardware we developed as shown in Figure 9. The $1/3$ Hz power samples are calculated using current and voltage values sampled at a high frequency. The voltage and current sampling frequency can be adjusted to control the noise levels of the power data. We chose a lower sampling rate (200Hz) to analyze the effect of high measurement noise on model performance. For example, the ground truth curve of Figure 10 shows the noisy data collected from refrigerator over 800 minutes. We only tested (M2) on custom data since performance of (M3) is expected to be superior. Table 3 shows the disaggregation results for kettle, microwave and



(a). (M1) - refrigerator (b). (M2) - refrigerator (c). (M3) - refrigerator

(d). (M1) - washing machine (e). (M2) - washing machine (f). (M3) - washing machine

(g). (M1) - microwave (h). (M2) - microwave (i). (M3) - microwave

(j). (M1) - kettle (k). (M2) - kettle (l). (M3) - kettle

(m). (M1) - dish washer (n). (M2) - dish washer (o). (M3) - dish washer

Figure 8: The disaggregation results on UK-DALE dataset for refrigerator, washing machine, microwave, kettle and dish washer using (M1), (M2) and (M3).

Table 2: The evaluation of the models

Metric	Model	FRDG	WASH	MWAV	KETL	DWSR
MAE	(M1)	24.45	10.77	11.24	20.55	24.20
	(M2)	21.10	12.00	8.42	16.82	35.80
	(M3)	**16.01**	**8.80**	**7.90**	**9.34**	**21.31**
SAE	(M1)	**0.082**	**0.020**	**0.385**	0.354	0.063
	(M2)	0.212	0.264	0.526	0.271	0.366
	(M3)	0.240	0.250	0.563	**0.242**	**0.013**
Recall	(M1)	**0.8590**	0.9246	**0.7060**	0.9678	**0.9140**
	(M2)	0.7880	0.8868	0.1480	0.9780	0.4827
	(M3)	0.8587	**0.9330**	0.1792	**0.9809**	0.9036
Precision	(M1)	0.8299	0.3369	0.2990	0.9021	0.2941
	(M2)	0.8119	0.3077	0.3881	0.9330	**0.7646**
	(M3)	**0.9205**	**0.6211**	**0.4478**	**0.9602**	0.5843
Accuracy	(M1)	0.8589	0.9780	0.9900	0.9986	0.9354
	(M2)	0.8245	0.9757	0.9942	**1.0000**	**0.9811**
	(M3)	**0.9041**	**0.9926**	**0.9947**	0.9994	0.9790
F1 score	(M1)	0.8441	0.4939	**0.4201**	0.9338	0.4450
	(M2)	0.7998	0.4568	0.2070	0.9550	0.5918
	(M3)	**0.8885**	**0.7458**	0.2560	**0.9705**	**0.7097**

(a) UK-DALE dataset

Metric	Model	FRDG	WASH	MWAV	DWSR
MAE	(M1)	-	-	-	-
	(M2)	23.49	40.18	14.45	5.12
	(M3)	**22.14**	**40.12**	**9.56**	**5.10**
SAE	(M1)	-	-	-	-
	(M2)	0.054	**0.387**	**0.254**	**0.126**
	(M3)	**0.042**	0.439	0.328	0.285
Recall	(M1)	-	-	-	-
	(M2)	**0.9413**	0.5894	**0.9891**	**0.7508**
	(M3)	0.9301	**0.5266**	0.7224	0.7169
Precision	(M1)	-	-	-	-
	(M2)	0.9243	0.8114	0.3210	0.6085
	(M3)	**0.9513**	**0.9848**	**0.8983**	**0.4808**
Accuracy	(M1)	-	-	-	-
	(M2)	0.9393	0.9774	0.9918	**0.9911**
	(M3)	**0.9476**	**0.9799**	**0.9985**	0.9872
F1 score	(M1)	-	-	-	-
	(M2)	0.9327	0.6828	0.4847	**0.6722**
	(M3)	**0.9406**	**0.6863**	**0.8009**	0.5756

(b) REDD dataset

refrigerator. From the high disaggregation and event detection performance, it can be concluded that the model is robust to measurement noise. Figure 10 shows an example disaggregation for refrigerator. The denoising effect of the model can be observed where the predicted power curve is less noisier than the actual power curve. The noise variance of the actual appliance level power data for the refrigerator shown in Figure 10 is around 21.4 whereas the corresponding value for the predicted appliance level power data is around 5.3.

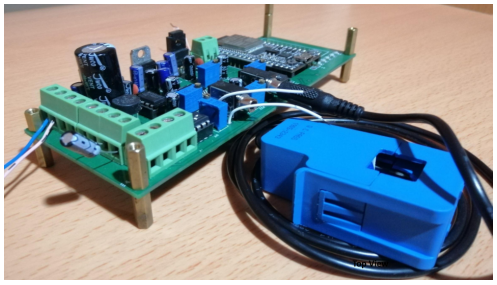


Figure 9: Power monitoring hardware.

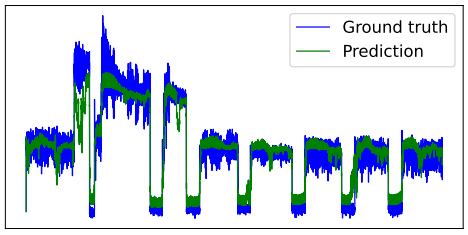


Figure 10: Example disaggregation for the refrigerator using custom data.

Table 3: The evaluation of (M2) for custom data

Metric	FRDG	KETL	MWAV
MAE	6.65	3.82	46.06
SAE	0.01	0.05	0.88
Recall	0.9839	0.9778	0.9839
Precision	0.9905	0.9910	0.9717
Accuracy	0.9797	0.9991	0.9985
F1 score	0.9872	0.9843	0.9777

6. Conclusion

We have combined sequence-to-short-sequence learning with a novel attention-based autoencoder architecture and have explored how the combination can be used to enhance both the accuracy and the real-time capability of NILM systems. We have empirically validated the superior performance of the proposed model by comparing it with a state-of-the-art model using real-world data. We have also used a custom dataset to evaluate our models using noisy data from which we demonstrated the robustness of the models. Detection of appliances with anomalous power consumption and real-time detection of events such as refrigerator door opening are possible future extensions of the work proposed in this paper.

References

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

[2] A. Ridi, C. Gisler, J. Hennebert, A survey on intrusive load monitoring for appliance recognition, in: Proc. 22nd International Conference on Pattern Recognition, 2014, pp. 3702–3707.

[3] A. Cominola, M. Giuliani, D. Piga, A. Castelletti, A. Rizzoli, A hybrid signature-based iterative disaggregation algorithm for non-intrusive load monitoring, Appl. Energy 185 (2017) 331–344.

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

[5] M. Baranski, J. Voss, Genetic algorithm for pattern detection in NIALM systems, in: Proc. IEEE International Conference on Systems, Man and Cybernetics, Vol. 4, 2004, pp. 3462–3468. doi:10.1109/ICSMC.2004.1400878.

[6] K. Srinivasarengan, Y. G. Goutam, M. G. Chandra, S. Kadhe, A framework for non intrusive load monitoring using Bayesian inference, in: Proc. Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 2013, pp. 427–432.

[7] H. Kim, M. Marwah, M. Arlitt, G. Lyon, J. Han, Unsupervised disaggregation of low frequency power measurements, in: Proc. SIAM International Conference on Data Mining, Vol. 11, 2011, pp. 747–758.

[8] J. Kelly, W. Knottenbelt, Neural NILM: Deep neural networks applied to energy disaggregation, in: Proc. 2nd ACM International Conference on Embedded Systems For Energy-Efficient Built Environments (BuildSys'15), 2015, p. 55–64. doi:10.1145/2821650.2821672.

[9] H. Shao, M. Marwah, N. Ramakrishnan, A temporal motif mining approach to unsupervised energy disaggregation: Applications to residential and commercial buildings, in: Proc. Twenty-Seventh AAAI Conference on Artificial Intelligence, 2013, p. 1327–1333.

[10] R. Bonfigli, E. Principi, M. Fagiani, M. Severini, S. Squartini, F. Piazza, Non-intrusive load monitoring by using active and reactive power in additive factorial hidden Markov models, Appl. Energy 208 (2017) 1590–1607.

[11] R. Jia, Y. Gao, C. J. Spanos, A fully unsupervised non-intrusive load monitoring framework, in: Proc. IEEE International Conference on Smart Grid Communications (SmartGridComm), 2015, pp. 872–878.

[12] J. Z. Kolter, S. Batra, A. Y. Ng, Energy disaggregation via discriminative sparse coding, in: Proc. 23rd International Conference on Neural Information Processing Systems, Vol. 1, 2010, p. 1153–1161.

[13] C. Zhang, M. Zhong, Z. Wang, N. Goddard, C. Sutton, Sequence-to-point learning with neural networks for non-intrusive load monitoring, in: Proc. AAAI Conference on Artificial Intelligence, Vol. 32, 2018.

[14] A. Harell, S. Makonin, I. V. Bajić, Wavenilm: A causal neural network for power disaggregation from the complex power signal, in: Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 8335–8339.

[15] G. Zhou, et al., Sequence-to-sequence load disaggregation using multiscale residual neural network, IEEE Transactions on Instrumentation and Measurement 70 (2021) 1–10.

[16] J. Liang, et al., Deep neural network in sequence to short sequence form for non-intrusive load monitoring, in: Proc. IEEE 3rd Conference on Energy Internet and Energy System Integration (EI2), 2019, pp. 565–570.

[17] S. Kiranyaz, et al., 1D convolutional neural networks and applications: A survey, Mechanical Systems and Signal Processing 151 (Apr. 2021).

[18] A. Vaswani, et al., Attention is all you need, in: Proc. 31st International Conference on Neural Information Processing Systems, 2017, p. 6000–6010.

[19] V. Piccialli, A. M. Sudoso, Improving non-intrusive load disaggregation through an attention-based deep neural network, Energies 14 (Feb. 2021).

[20] T. Luong, H. Pham, C. D. Manning, Effective approaches to attention-based neural machine translation, in: Proc. Conference on Empirical Methods in Natural Language Processing, 2015, pp. 1412–1421.

[21] J. Kelly, W. Knottenbelt, The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes, Scientific Data 2 (Mar. 2015).

[22] J. Kolter, M. Johnson, REDD: A public data set for energy disaggregation research, Artif. Intell. 25 (2011).

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

[24] C. Dinesh, S. Makonin, I. V. Bajic, Incorporating time-of-day usage patterns into non-intrusive load monitoring, in: Proc. IEEE Global Conference on Signal and Information Processing (GlobalSIP), 2017, pp. 1110–1114.