NILM Methods for low-frequency smart meter data

Surya Venkatesh Pandiyan

Thesis to obtain the Master of Science Degree in

Energy Engineering and Management

Supervisor: Prof. Carlos Augusto Santos Silva

Examination Committee

Chairperson: Prof. Duarte de Mesquita e Sousa
Supervisor: Prof. Carlos Augusto Santos Silva
Member of the Committee: Dr. Henrique Ramalho Monteiro Latourrette Pombeiro

December 2020

ACKNOWLEDGEMENTS

Foremost, I am grateful for the journey I had these two years as a foreign graduate student which I believe made me mature. In this journey, I met some great peoples, found new friends, experienced a new culture, gone through different emotional situations. As they say, a journey must end one day, I would like to acknowledge a few peoples who made this journey wonderful and possible to end it with this successful thesis.

I would like to thank this wonderful person who has been a major part of this journey, my supervisor Dr. Carlos Santos Silva. he did not hesitate when I asked him research opportunity during summer break even though I did not have any machine learning experience. He treated me as a colleague from day one. he appreciated my work which I have done with another professor even when he did not have to. His small appreciations have made me push further. He gave me all the freedom I wanted with this thesis and also stepped in when I needed guidance and advice. Not only for the thesis, but he also took the time to help me with my future endeavors. One can not ask more from the supervisor which is why I am highly grateful to him.

I would like to thank IST for this course and all the amazing professors who are part of it for teaching me new knowledge and skills and taking me one step closer to my goal.

I would like to thank two of my dear friends Mr. Arvind and Mr.shemin. Arvind is the one who took more interest in my emotional well being than myself. He is always encouraging, pushing me to improve myself. Shemin is always enthusiastic and optimistic and it is a little contagious. I am grateful for their friendship and will cherish it forever.

Finally, last but not least, my parents without them this would not even be possible. I am grateful for their love, care, support, and trust in me. I am forever indebted to them.

ABSTRACT

In this work, we proposed two new models variational Autoencoder (VAE) based classifier and statistical rules-based temporal classifier for very low-frequency unsupervised Non Intrusive Load Monitoring (NILM) and tested the state of art NILM method (Hidden Markov Models (HMM) with general priors) proposed for 10s data interval for 10min data interval. We adopted the HMM model for Factorial Hidden Markov Model (FHMM) to disaggregate the set of appliances from total consumption. We also proposed and tested a novel post-processing method based on K-Nearest Neighbor (K-NN), bottom-up approach, and similar time window methods. The model verification shows that these three models are good enough only for the appliances water heater and space heater for real-world deployment. Our two methods VAE and statistical rules-based methods shows superior performance than and similar performance to the state of art HMM method. Our detailed analysis shows that some of the issues of all three models can be solved by having already proposed sophisticated noise filters, data preparation, dependent appliance models, and post-processing. Our novel post-processing has proven to improve the recall value and F1-score even when its value is less than 0.1. This method can be integrated with any probabilistic NILM models. Our model validation shows that for very low frequency, The model cannot be completely non-intrusive and as well as high performing. The results indicate NILM needs technological innovation and business innovation to gain trust and penetrate the market.

Key-words: Non Intrusive Load Monitoring, smart grid, unsupervised learning, Hidden Markov Model, Variational Autoencoder , temporal classifier, K-Nearest Neighbor, bottom-up approach, similar time window, post-processing.

RESUMO

Neste trabalho, propusemos dois novos modelos de classificador variational autoencoder (VAE) e classificador temporal baseado em regras estatísticas para Desagregação de carga não intrusiva(NILM) não supervisionado de frequência muito baixa e testamos o método NILM de última geração (Hidden Markov Model (HMM) com priors gerais) proposto para intervalo de dados de 10s para Intervalo de dados de 10min. Adotamos o modelo HMM para Factorial Hidden Markov Model (FHMM) para desagregar o conjunto de eletrodomésticos do consumo total. Também propusemos e testamos um novo método de pós-processamento baseado em K-Nearest Neighbor (K-NN), abordagem ascendente e métodos de janela de tempo semelhantes. A verificação do modelo mostra que esses três modelos são bons o suficiente apenas para os aparelhos aquecedor de água e aquecedor de ambiente para implantação no mundo real. Nossos dois métodos VAE e métodos baseados em regras estatísticas mostram desempenho superior e semelhante ao método HMM de última geração. Nossa análise detalhada mostra que alguns dos problemas de todos os três modelos podem ser resolvidos com a proposta de filtros de ruído sofisticados, preparação de dados, modelos de aparelhos dependentes e pós-processamento. Nosso novo pós-processamento provou melhorar o valor de recall e a pontuação F1, mesmo quando seu valor é inferior a 0,1. Este método pode ser integrado a qualquer modelo NILM probabilístico. A validação do nosso modelo mostra que, para frequência muito baixa, o modelo não pode ser completamente não intrusivo e com alto desempenho. Os resultados indicam que a NILM precisa de inovação tecnológica e inovação empresarial para ganhar confiança e entrar no mercado.

Palavras-chave: Desagregação de carga não intrusiva, rede inteligente, aprendizagem não supervisionada, Hidden Markov Model, Variational Autoencoder, classificador temporal, K-Nearest Neighbor, abordagem ascendente, janela de tempo semelhante, pós-processamento.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

At present, government and energy companies around the world have three major concerns: Depleting fossil reserves, securing sustainable energy supply, and the effects of climate change [1]. To tackle these issues, governments and various companies proposed and working on electrification (e.g. transport systems and heating systems) and electricity production through renewable energy. This in turn for sure, increases the electricity demand. In addition to that, an increase in the global population and developing countries' aims like rural electrification, and economic growth will further increase the electricity demand. It is, therefore, necessary to find methods by which we can tackle such demands without negative effects.

In the stated policies scenario, the global electricity demand grows at 2.1% per year to 2040. This increases the total electricity share in final energy consumption from 19% in 2018 to 24% in 2040. According to the international energy agency, by sector, residential electricity demand is expected to increase by 83% of the 2018 demand by 2040. This is the second-highest increase next to the transport sector but transport sector contribution to total demand is expected to be very less by 2040 (11%). The second major contributor to total electricity demand is the residential sector (23% in 2018 and 26% by 2040) next to the industrial sector (34.5% in 2018 and 33% by 2040). The flexibility needs are expected to increase by 42 % in the European Union. All the above-stated numbers are based on the stated policies scenario reported in the International energy agency analysis report [2].

So, it is important to optimize house energy demand/usage for cost reduction or energy reduction, or both. In this thesis, we focus on one such method to optimize house energy demand for cost reduction. The next section introduces the problem statement.

1.1 Problem statement

One of the cost reduction methods which benefits both producers and consumers are load deferral during peak demand [3]. Consumers would benefit from low peak prices and producers will benefit from unneeded additional investment for increasing production capacity or energy storage capacity to meet the peak demands. Load deferral is the concept of rescheduling appliances from peak demand period to low demand period.

To implement this technique, there is a need to know appliance power consumption in real-time. The research problem addressed in this work is how to find the appliance power consumption in real-time both economically and less intrusively?

There are many ways to find appliance power consumption in real-time. One method is Deploying one smart meter for each appliance which can communicate in real-time. This method is undesirable due to higher manufacturing and installation cost, inconvenience, and maintenance[4].

Another method is using a special smart meter for aggregate consumption reading which provides high-frequency readings of consumption. High-frequency readings provide more electrical features that help to find appliance consumption easily. This eliminates the need to have individual meters for each appliance. However, this method is also undesirable because of higher manufacturing costs [5], and replacing already deployed low-frequency smart meter is a pain for electricity companies.

Another alternative approach would be to use the already deployed low-frequency smart meters. These smart meters provide aggregate consumption at regular intervals usually in minutes although some meters provide second-based readings. These smart meters are designed for billing. The consumption interval reading greater than one minute is not ideal for low power /multi-state devices disaggregation purposes [6]. It contains very less useful features for disaggregation. Portugal's smart meters are expected to register readings at least every 60 seconds and communicate with the data center every 15 minutes[7].

To deal with very low-frequency readings (above 1 minute), three methods are widely researched. First, using only the available aggregate consumption data to disaggregate but accuracy would be less. Second, collect the past data for the house and use these data for future disaggregation but the need to deploy meters to each appliance is discouraging. Third, data collected from other houses are used to disaggregate for the house of interest (transfer learning). The third approach is gaining interest in recent years for very low-frequency disaggregation as more datasets are released for research purposes and it balances the cost and accuracy when compared to the other two methods. This thesis focuses particularly on the third approach for practicality reasons. Moreover, the COVID-19 pandemic situation might make the data collected before useless. The IEA has shown that pandemic shifted the consumption pattern in the industries, residential sector, etc [8].

To summarize, the solution requires the following traits:

- Appliance load monitor: The method should estimate the appliance power consumption
- Non-intrusive: The method should use only a single point of data collection for all appliances in the house and collect minimum information from household members.
- Low-frequency data: The method should be suitable for appliance monitoring given aggregate consumption data at 5-15 minutes interval.
- Unsupervised: The method should not require training data for appliance monitoring from the house of interest.

1.2 Potential challenges

There are several challenges for deploying Non Intrusive Load Monitoring (NILM) solutions which can be broadly classified as technical and social challenges. In this section technical challenges are discussed first, and then social challenges related to NILM are discussed.

NILM relies on the features detected from the smart meter reading to disaggregate total energy consumption to appliance consumption. However, in a household, multiple appliances operate simultaneously. As a result, there will be an overlap of multiple appliance features. Such overlap not only obscures the low power appliance features but also merged features produces a completely new feature [9]. As the number of appliances operating simultaneously in a house increases so does the complexity of the disaggregation.

Multiple variables affecting the energy consumption pattern of the appliance is another technical challenge. For example, different households contain different appliances for the same purpose (electric or gas heating for water heating and cooking). Another such problem would be the difference in energy consumption of the same appliance type (different size water heaters) and with the same capacity (old refrigerator vs new refrigerator). Furthermore, multiple working modes exist and the user’s preference for an appliance (microwave and washing machine) adds further complexity. A higher number of appliances exist in households and multiple appliances with the same operating principles (kettle and toaster) also affect the disaggregation [6].

NILM also must deal with the noise from various sources (smart meter noise, Appliance noise due to the supply voltage. Etc). The above-mentioned technical challenges cause a huge difference in the performance of NILM between households. So, it is a challenge to build NILM solutions that perform satisfactorily in all conditions.

In addition to technical challenges, some social barriers are affecting the deployment of the NILM system. The major concern is privacy intrusion. From the disaggregated data, there are multiple inferences can be made of household members’ habits and occupancy [10]. Occupants might not prefer that. However, there are ways around to overcome this social barrier like performing the disaggregation in the house itself, not letting the data leave the house, or giving options to the occupants to opt-in for the NILM system [9]. The next section summarizes the contribution of this thesis addressing some of these challenges.

1.3 Research contributions

Our research objective for this thesis is as follows:

To examine the performance of Popular unsupervised NILM methods proposed for low-frequency data(1s-1min) but not tested for very low-frequency data (5min-30 min). Test the practicality of these methods with less intrusion and/or survey. Develop new methods or adapt the existing methods for very low-frequency data to improve performance. study the practicality of implementing peak demand response and find suitable appliances for it.

This thesis research contributes to the NILM field as follows:

1. Evaluating the Hidden Markov mode l(HMM) with the appliance general model proposed in [11] for very low-frequency data and adapting it for the Factorial Hidden Markov model (FHMM) to find the contribution of a set of appliances to total energy consumption. In this thesis, only a set of space heaters consumption contributed to total electricity consumption is evaluated but generalizable to other sets of appliances.
2. Developed an autoencoder based disaggregation solution. As for what we learned from the literature study, this is the first attempt to use Auto-encoder based classifier for NILM. We found Auto-encoder performs reasonably and viable option for disaggregation.
3. Created an unsupervised method based on the supervised temporal classification method proposed in [12] and evaluated its performance.
4. Finally, a novel post-processing system based on the K-Nearest Neighbor (K-NN) algorithm and bottom-up approach is proposed to improve the disaggregation results of any NILM method. This method has proven to be a viable option to deal with noise and multiple appliances operating unexpectedly at the same time. We also show that this method performance relies on a top-down approach (NILM methods) performance. The next section describes the structure of this thesis.

1.4 Structure of the thesis

The remaining chapters of the thesis are structured as follows:

Chapter 2 provides information on relevant researches and backgrounds related to NILM. First, Short descriptions of types of existing solutions, applications, working scenarios, datasets are presented. Later, low-frequency methods are discussed in particular focus to the methods relevant to this thesis.

Chapter 3 presents the results of data analysis on the train houses and test houses. These results provide information regarding appliance usage like time of use, weekday of use, monthly variations, percentage of total consumption, Noise from other appliances when it is operating.

Chapter 4 describes the theoretical background of the Algorithms used in this thesis. First, the Hidden Markov model using the general appliance model is explained followed by its adoption FHMM for a set of appliances. Second, how an autoencoder based classifier can be adopted for NILM is discussed. Third, a description of how a supervised temporal classification algorithm is adopted into an unsupervised method is given. Finally, a new contribution to NILM, bottom-up-approach-based K-Nearest Neighbor for post-processing is elaborated.

Chapter 5 presents the results from the methods tested in this thesis followed by detailed discussions of our findings.

Chapter 6 presents the conclusions from this work and discusses the possible future extension of this work with the specific intention of improving accuracy and practical deployment in the real world.

The next chapter presents the state of art for load monitoring.

2 Background and Literature review

This chapter describes various approaches to identify appliance power consumption and explains the reasons why, currently, NILM is a promising approach for it. This chapter starts with intrusive based approaches and explains the issue of poor scalability. Second, we describe the NILM steps and relevant research for each step. Finally, We found that the most popular low-frequency non-event-based methods proposed are in the frequency range of 1s-1min but in some countries, the smart meters deployed are in the frequency range above 1min. it is needed to validate these approaches for very low-frequency data and it is what explored in this thesis and proposed some new methods. Finally, we give some short descriptions about performance metrics, appliance types, applications of NILM, working scenarios, and popular datasets widely used for NILM research.

2.1 Intrusive approaches

Intrusive approaches involve the deployment of sensors to find the appliance power demand. There are two methods involved, direct method and indirect method. The direct method measures the electrical characteristics of the appliance from which power demand is identified. In the indirect method, sensors measure nonelectrical characteristics from which appliance power demand is inferred. Short discussions about both methods are presented below.

2.1.1 Direct monitoring

We discuss three different forms of direct monitoring under this section and explains why each one is unsuitable.

2.1.1.1 Electrical sub-metering

This method involves deploying power sensors to each device in the house. These sensors are usually plug-in meter or clamp-on meter. These provide accurate measurements of appliance power demand. However, installation cost, time requirements, and maintenance are cited as the major drawbacks of this method.

2.1.1.2 Smart appliances

Smart appliances are capable of self-reporting their energy consumption. Smart appliances with wireless communication modules can send the data to the central hub in the house or the cloud. However, this approach also has major drawbacks. All the old appliances need to be either replaced or retrofitted. It is expected that IOT connected appliances need at least 20 years to capture the substantial market[13]. Besides, there is a need to be all appliances from the same company or all companies agree to sell the data to a single entity. Even though it is promising, there is a long road ahead for this method to become a viable option.

2.1.1.3 Electrical probing

Electrical probing is the process of sending a signal into an electrical circuit and inferring results based on the returned signal[14]. This method adds interference to the electrical circuit which affects the power quality adversely. So electrical probing was not widely researched for appliance load monitoring. The next section is about indirect methods.

2.1.2 Indirect monitoring

In this section, we discuss three forms of indirect monitoring methods and their disadvantages.

2.1.2.1 Appliance tagging

In this method, when an appliance is turned on or off, a tag will send a signal indicating the appliance has been turned on or off. However, this method requires modification to the appliance. Installation and customization cost, maintenance, and time-consuming are discouraging factors of this method. One such example was demonstrated and presented in [15], using Radio-Frequency Identification (RFID) signal to uniquely identify appliances.

2.1.2.2 Ambient sensors

Ambient sensors like audio, light, and temperature sensors could be used in the household to measure variables other than electrical ones to monitor human behaviour and appliance operation [16]–[18]. This requires the intrusive installation of many sensors and complex algorithms. This method is not suitable for scale-up. Also, it is difficult to model human behaviour without any information or survey from household members. This method is suitable when NILM is deployed along with home automation or assisted living applications.

2.1.2.3 Conditional demand analysis (CDA)

Unlike other methods, CDA does not require the installation of a special device to disaggregate energy. Alternatively, it relies on other data about consumers, household type, and weather. This requires a large survey to be conducted. Based on this data, a multivariate regression analysis technique [19] can be used to find appliance power demand. A survey is a form of social intrusion and the reliability of such data discourages this method. Sometimes, people do not aware of their habits which itself changes periodically.

Clearly, the above-mentioned methods are not suitable for appliance monitoring, so we move on to the NILM from the next section. It should be noted though that partial deployment of these methods (ambient sensors, CDA, smart appliances) could assist NILM to better disaggregate total energy. We are not exploring that option in this thesis.

2.2 Non-intrusive monitoring

The process of NILM involves the following steps as shown in Figure 1. This section aims to provide a review of these steps and various approaches for these steps found in the literature.

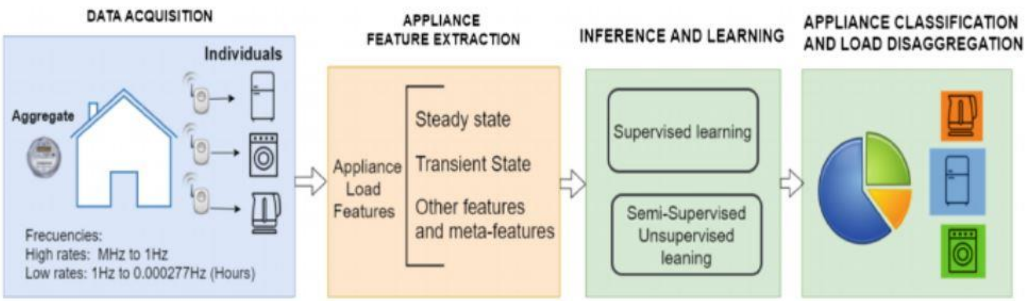

Figure 1: General pipeline of NILM *[20]*

2.2.1 Data acquisition

In NILM, data is collected at a single sensing point. As already mentioned most smart meters rolled out by electrical companies are less than 1hz [21]. The data collected are usually current, voltage, active power, reactive power, and some more data. However, there are commercial companies that provide NILM as a solution that relies on special hardware. This hardware usually collects data at high frequency. The type of data collected, and techniques employed for that will have major consequences in terms of applications NILM can tackle performance, and cost [21]. A technical review has been made on smart meters based on 41 commercially available smart/energy meters in [22]. The parameters sensed by these meters are shown in Figure 2 taken from [22]. for certain parameters like harmonics, their sample frequency requirement should be met. However, the downsides of these high-frequency meters are cost and bulky storage requirements. In Figure 3, the relation between disaggregation granularity and the sampling frequency is shown. We can see that in a low-frequency approach only the most significant appliances can be monitored reliably.

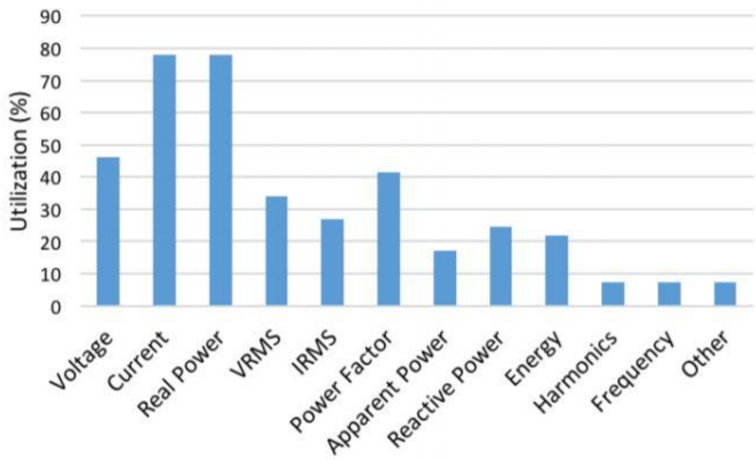

Figure 2: Parameters collected by the smart meters*[22]*

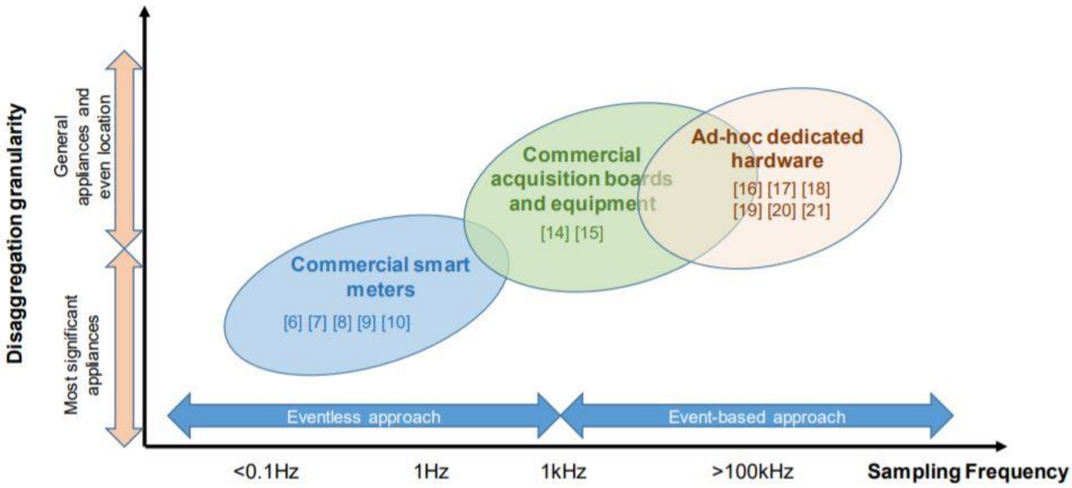

Figure 3: Approaches and their performance for different sensors*[21]*

Having explained the data acquisition we move on to the next step event detection in the following section.

2.2.2 Event detection

We consider approaches proposed for the frequency above 1hz as high-frequency NILM. High-frequency NILM mostly involves event-based approaches due to rich features available due to high frequency. Although, some non-event methods can also be found in the literature for medium-range frequencies (1hz to 1Khz). Event-based approaches rely on event detection. When a signal changes from one steady-state to another, it is considered an event.

For event detection, a high sampling rate is required [21]. The approaches taken in event detection can be grouped under three categories:

- Expert heuristics
- Probabilistic models
- Match filters

Expert heuristics is a simple method. The basic idea is scanning the time series data and look for changes above the threshold level between two consecutive samples[20]. Different methods were approached depending upon the intended appliances to detect and sample rate. For example, different features like active power, reactive power, harmonics are combined to detect appliances with similar active power change[23].

Table 1:various features for NILM found in the literature

Feature category	References	Low frequency			High frequency			
		Steady		Transient	Steady			Transient
		ΔP ΔQ	I, V	Macro	FFT	Shape	STFT FFT	WT
P–Q plane	[14][24]	√						
P–Q plane and macroscopic transient	[25][26]	√		Shape				
P–Q plane and macroscopic transient and harmonics	[27]	√		Duration	Har			
	[28]	√		Shape	Har			
P–Q plane and harmonics	[29]	√			Har		Har	
	[30]	√					Har	
	[31][32]	√					Har	
	[33]	√			THD			
Real power only	[34]–[39]	ΔP						
Low-frequency P, I and V based features	[40]	ΔP	RMS, max					
	[35]	√ PF						
Spectral envelope (SE)	[41]–[43]						SE	
Wavelets	[44]–[48]							W_{coef}
	[49]							W_e
P–Q plane and wavelets	[50]	√			THD			W_{coef} , tr , W_e
	[51]	√						W_{coef}, W_e
	[52]	√						W_{coef}
Shape features	[53]–[55]					VI–traj		
Raw waveforms	[56]					i(t)		
Nonactive current	[57]					i_f (t)		
	[58]					VIf–traj		
Combination	[59]	√			Har	i(t), p(t), VI–traj		
	[60]	√			Har	i(t), VI–traj		
	[61]	√	I_{cr}		Har, THD	p(t), VI–traj	Har	

ΔP: step changes in real power; ΔQ: step changes in reactive power; RMS: root-mean-square of current and voltage; max: Maximum value of current and voltage wave-forms; I_{cr} : current crest factor; PF: power factor, the ratio of real to apparent power (p/s); Har: coefficients of fast Fourier transform (FFT); THD: Total harmonic distortion; VI–traj: voltage–current trajectory; VIf–traj: voltage–nonactive current trajectory; i(t): instantaneous current waveform; i_f (t): instantaneous nonactive current waveform; p(t): instantaneous power waveform; W_{coef} : coefficients of wavelet transform; W_e : energy of wavelet transform coefficients; t_r : transient response time based on wavelet coefficients.

The probabilistic model on the other hand uses the probability of an event occurring to do the event detection. The most popular statistic models used are Generalized likelihood Ratio[62], Goodness of fit [63], cumulative sum[64], kernel Fisher discriminant analysis[65].

Match filters use filtering techniques to detect events by comparing the known appliance transients in the aggregate consumption signal. More reviews on all three methods can be found on [23][4][21][62][20].

Recently decision trees and long short-term memory (LSTM) [21] are also tried for event detection. Since event detection is not relevant to this thesis, we move on to feature extraction.

2.2.3 Feature extraction

Table 2: Advantages and shortcomings of steady-state features

Steady-State Methods	Features	Advantages	Shortcomings
Power Change [14], [28], [66], [67]	Steady State Variation of Real and Reactive Power, P; Q	High-Power Residential Loads can easily be identified, Low-sampling rate requirement,	Low power appliances overlap in P-Q plane, Poor performance in recognizing Type-II, III and Type-IV loads.
Time and Frequency Domain Characteristics of VI Waveforms [30], [40], [59], [68]–[71]	Higher-order Steady-State Harmonics, Irms, Iavg, Ipeak, Vrms, Power factor	Device classes can easily be categorized into resistive, inductive and electronic loads	High sampling rate requirement, Low accuracy for Type-III loads, overlapping features for consumer electronics of Type-I and II category, unable to distinguish between overlapping activation events
V-I Trajectory [72][53]	Shape features of V-I trajectory: asymmetry, looping direction, area, the curvature of mean line, self-intersection, slope of middle, segment, area of segments and peak of middle segment	Detail taxonomy of electrical appliances can be formed due to distinctive V-I curves	Sensitive to multi-load operation scenario, computationally intensive, smaller loads have no distinct trajectory patterns
Steady-State Voltage Noise[18][73]	EMI signatures	Motor-based appliances are easily distinguishable as they generate synchronous voltage noise, Detection of simultaneous activation events, Consumer appliances equipped with SMPS can be recognized with high accuracy	Sensitive to wiring architecture, EMI signatures overlap, not all appliances are equipped with SMPS

After event detection, the next step is feature selection which is used for load identification. In an event less approach, event detection is skipped, after data collection, feature selection is the next step. Feature available and selection depends on the sample rate available and the method used for load identification.

Appliances signatures are the key to detect and classify appliances. Signatures can be classified into steady-state signatures and transient state signatures. Steady-state signatures give us the signatures of the devices when operating in the same state. Transient state signatures give us the details when the device change from one state to another. Sometimes, steady-state signals alone cannot detect and classify devices. Transient state signatures can be used along with steady-state to improve classification accuracy.

An extensive review has been done in[74] on feature selection and algorithm for feature selection. The data from that study are shown in Table 1.Another great source for feature review is [75]. The data presented in that study are shown in Tables 2 and 3. The details of the algorithm and features are skipped in this report because the event less approach is the motive and there are very few features that can be extracted. Real power and reactive power are the most used steady-state features in low-frequency algorithms. The advantages and shortcomings of these features can be seen in tables 2 and 3.

Table 3: Advantages and shortcomings of transient state features

Transient Methods	Features	Advantages	Shortcomings
Transient Power [76][30][77][50][42]	Repeatable transient power profile, spectral envelopes	Appliances with the same power draw characteristics can be easily differentiated, Recognition of Type I, II, III loads	Continuous monitoring, high sampling rate requirement, not suitable for Type IV loads
Start-Up Current Transients[28][25][50]	Current spikes, size, duration, shape of switching transients, transient response time	Works well for Type I and II loads, distinct transient behavior in multiple load operation scenario	Poor detection of simultaneous activation deactivation of sequences, unable to characterize Type III and IV loads, sensitive to wiring architecture, appliance specific
High-Frequency Sampling of Voltage Noise[18][78]	Noise FFT	Multi-state devices, consumer Electronics with SMPS	Appliance specific, computationally expensive, Data annotation is very hard

Apart from these, non-traditional features[79]–[81] like time, day of use, weather data for HVAC systems, the correlation between different appliances, occupant's behaviour are also been proposed. Sometimes combining the features available also proposed. 55 steady-state features and 23 transient state features are combined and reduced to 20 features[74].

2.2.4 Load identification

The next step after feature extraction is load identification. Identifying what appliances are contributing to the aggregate consumption signal is load identification. In some eventless approaches, feature selection is integrated into load identification algorithms like HMM, Neural network. A Myriad of algorithms has been proposed for this last step. only important and recently developed approaches are presented in this section.

2.2.4.1 Optimization method

The load disaggregation can be modelled as an optimization problem. The principle is, it compares the extracted features of the unknown load to the known loads present in the appliance database and tries to reduce the error to find the closest possible match. It is a very complex approach as an algorithm must consider possible combinations of appliances in the database that produces the composite load. If the historical data is available, the database can be created for optimization and run as supervised solving. If historical data is not available, the problem becomes more complex and a general model of the appliances needs to be created and run as an unsupervised problem. The optimization method can gain high disaggregation accuracy however computational burden for the optimization method is so high [82]. When the number of appliances increases the performance decreases[16].

Different optimization methods can be used for solving the disaggregation problem. The examples are hybrid programming[83], genetic programming, segmented integer quadratic constrained programming. In [84] aided linear integer programming is introduced to improve the integer programming-based disaggregation. in [85] the feasibility of an evolutionary algorithm to solve NILM is verified. in [86] bird swarm optimization is evaluated for load disaggregation. The load signature used in the study is the load pattern of the appliances.

2.2.4.2 Supervised learning/pattern recognition methods

It is the most researched method in NILM. A simple clustering-based approach is one type of method proposed by hart [14]. In this method cluster of appliances is developed with known loads of the appliances. during analysis, the extracted feature is compared with the known cluster to classify. The major drawbacks of this method are failure to cluster appliances with overlapping P-Q features and sensitivity to power drifts. Various improvements have been proposed for these problems in [31][67][70].

Bayesian classifiers[66][87][88] is another main approach based on probability theory in which it is assumed that the state of the appliances is independent of each other. However, in the real world, it is not true. For example, the use of PlayStation and TV. Even though the Bayesian approach shows promising results in controlled lab experiments, the concern expressed is, for high accuracy, the appliances should have a clear set of features.

Support Vector Machines(SVM)[35] is another popular method that uses the optimal line separation method. The optimal line separation method is the process of finding the line separating the classifications with maximum margin in which none of the training data falls in. some of the features of SVM are,1. SVM performs worse with large data. 2. SVM does not converge to local maxima. 3. Input dimensions will not affect computational complexity. Higher accuracy of 99% is shown in[89] and also at lower frequency samples the accuracy of SVM studied in [90] is 59%.

K-nearest neighbor approach [64] is based on distance function in which classification is made when unlabelled data is nearest to K nearest labelled data. Implementation and computation are simple. The accuracy of 99% is proven in [35] with a 5-NN classifier. The accuracy of 85% is shown in [32] with 1 NN classifier with Euclidean distance measurements.

Hidden Markov Model (HMM) and artificial neural networks (ANN) are also a popular choice due to their ability to temporal data and state change information. HMM is based on the Markov chain. the complexity of the HMM increases exponentially with an increase in appliances, limiting its applicability in the real world[91]. various types of HMM has been proposed to solve this problem[92][93].

Neural Network is a type of machine learning algorithm inspired by how neurons work in our brains. For NILM various architectures have been tested, Multilayer perceptron[50], Convolutional Neural Network (CNN)[94], long short-term memory(LSTM), Recurrent neural network (RNN), Auto-encoders. Accuracy of 95% is shown in the study[95] with ANN. An accuracy of 84% is shown in the study [96] with CNN and the accuracy of 80% is shown in the study [96] with LSTM. In the study proposed by Kelly[97], the NILM problem is treated as to separate a clear signal from the noisy signal, and denoising Auto-encoders were used to separate appliance signal by considering other appliance signals as noise. In that study, an F1 score of 55% is achieved. In recent years more deep learning study for NILM has been published.

In addition to that, restricted Boltzmann machine, conditional random fields[98], committee decision mechanisms[99] which combine different algorithms, decision trees[100] are other explored supervised methods.

2.2.4.3 Unsupervised learning

Even though supervised learning proves higher accuracy, the cost to collect data for training is a hindrance for real-world deployment. Unsupervised learning does not require training data.

Blind Source Separation (BSS) is used to solve NILM in [101]. BSS is a technique of separating source signals from aggregate signals with very little or no information known of the source. The steady-state features change in active power and reactive power is used to cluster the appliances. Each cluster is assumed to be a linear combination of multiple appliance sources. Matching pursuit is used for source reconstruction. Matching pursuit minimizes the error to find the closest possible cluster to aggregate load. Another example of BSS is using discriminative sparse coding [102] for load disaggregation.

Table 4: The performance details of various unsupervised techniques.

Contribution	Technique	Dataset	Signature category*	Number of appliances	Evaluation Criteria	Average Performance
[103]	FHMM	REDD	S	7	P - R - f_1	82.7% - 60.3% - 71.29%
[104]	HDP-HSMM	REDD	S	5	TECA	81.5%
[79]	CFHSMM	private	S	4-10	f_1	83.98%
[105]	HMM	REDD	S & T	5	D	70.6%
[106]	Optimization HMM	REDD	S	6	RMSE	0.002
[107]	DTW	REDD	S	6	P - R - f_1	91.24% - 81.77% - 86.16%
[108]	MSCS	BLUED	S & T	14	D	92%
[109]	Linear discrimination	private	S & T	16	Acc - P - R - f_1	81.56% - 78.93% - 63.81% - 70.57%
[110]	STMF	REDD	F	6	RMSE - DE	0.0631 - 0.0007
[81]	Contextual Convex	Pecan Street Sample	F	3	MAE	0.1889
[101]	Clustering BSS	private	S & F	18	Acc	69.3%

*Legend: F for full signal, S and T for steady and transient state signature category respectively.

Motif mining is studied in [111] which utilizes power cycles of individual appliances to disaggregate load. This method will not work for multistate appliances.

In unsupervised learning, the most researched model is HMM. This is due to the non-requirement of event detection. this is suitable for low-frequency samples as event detection is very difficult or not possible. In NILM, the Factorial Hidden Markov Model[112][103] is popular where observation is the sum of the output from each Markov chain. Conditional Factorial Hidden Markov Model (CFHMM) allows other features like time, dependencies between appliances. The Factorial Hidden Semi-Markov Model (FHSMM) allows an alternative probability density function to model the state occupancy duration of appliances. the merger of CFHMM and FHSMM has also been proposed[79]. In the study [103] the combination of additive HMM and difference HMM (DHMM) is used. Difference HMM was used to find the load difference between t-1 and t. they achieved an accuracy of 71%.

In the study [11] generic model of appliances was used along with Difference-HMM to do load disaggregation. The generic model was tuned for every house using aggregate signal alone.

In addition to this, dynamic time warping (DTW)[113], source separation via tensor and matrix factorizations, unsupervised Bayesian network [114]are some of the other approaches in unsupervised learning. An extensive review of the unsupervised techniques can be seen in[115]. The performance details of various unsupervised techniques are shown in Table 4.Table 4 is based on review work done in [115].

2.2.4.4 Signal processing and other approaches

Graph signal processing (GSP) is another powerful tool for data mining and signal processing. In the study [116], the GSP is used to solve NILM and proved to have a competitive performance with HMM models and it does not require a training Phase. Electromagnetic interference[66], wavelet design[89], power spectrum [117] based approaches are some other approaches studied in NILM.

Table 5: Details and performance of different NILM algorithm

Learning Algorithm	Features St [a]/Tr [b]	Accuracy %	Training S [c]/U [d]	Online/ Offline	Scalability	Appliance Types
SVM [18], [29], [69], [118]	B [e]	75–98	S	Online	Yes	I, II, III & IV
Bayes [66], [80], [118]	St	80–99	S	B	No	I&II
HMM [79], [103], [104]	St	75–95	B	Offline	No	I&II
Neural Networks [29], [40], [119]	B	80–97	S	Online	Yes	I&II&III
KNN [14], [120], [121]	B	70–90	S	B	Yes	I&II
Optimization [37], [39], [56], [59]	St	60–97	S	Offline	No	I&II
[a] Steady-State [b] Transient [c] Supervised [d] Unsupervised [e] Both.						

The data for Table 5 is collected from [75]. It is unfair to compare models based on the results published in the literature. Different studies use different datasets, different sample rates, different features. To compare the models should be evaluated with common datasets with common features and with common metrics. The data published in [21] consists of lists of various algorithms tried for NILM and their contribution, features used, sampling type.

2.3 Literatures adopted in this thesis

After an extensive literature review, considering multiple approaches have been taken and also many low-frequency algorithms are for sampling period between (1s-1min), we narrowed it down to four methods that are popular, capable to adapt to very low-frequency samplings, and suitable to the requirements we listed above.

As we mentioned before in [11] the generic framework has been proposed to create a general HMM model for the appliance with available data from other houses and use it to identify appliance properties at a particular house. They also showed how to tune the general model to a particular house in an unsupervised way. For disaggregation they proposed DHMM for each appliance and, they subtracted the disaggregated energy from the total before disaggregating the next appliance to maintain the constraint total consumption is the sum of appliances consumption. They also claimed this framework can be adapted to models other than HMM. This work had a great impact on the NILM community, but this work was validated for the sampling period of 10s. This work can be adopted for very low frequency and also capable of satisfying the requirements. This is the reason we adopt this work in the thesis. This work was also adopted in [106] where they proposed the optimization step as the final step instead of subtracting disaggregated value from total value to overcome the disadvantage that wrongly detected appliance value will affect all other appliance's value also.

Next to HMM, Neural network-based approaches are one of the popular methods for NILM. Usually, neural network-based methods are used in supervised setting but also some of them tested their model to the houses not seen beforehand [122]–[125], some of them explored the possibilities and capabilities of neural network for transfer learning [126], [127] in NILM context. We were surprised to find that Auto encoder-based classification was not tested for NILM as far as we know. Autoencoder based classification [128]–[130] is quite a popular technique for classification due to its ability to learn features as it is forced to compress data without losing its quality and also have been proposed for source separation problems [131]–[133]. The denoising autoencoder proposed is not a classification algorithm but a filter algorithm. Denoising autoencoder is trained in a supervised manner and this will not satisfy our requirements. We adopted this autoencoder based classification and proposed a new procedure for NILM.

The approach based on propositional concept learning applied to NILM in [12] is different from the usual classification techniques explored for NILM. It is a supervised technique. The step involves extracting sub-events from the training instances and by feature construction, the problem is represented in a way with these features for subsequent learning. These features represent temporal variation in the instance. For example, energy variation between time steps or distance between the current time step and local maxima would tell us something about the appliance in operation. This method also allows domain knowledge and background to be included. In [12] they were able to achieve high performance. We decided to explore this technique due to its performance and uniqueness and we made it unsupervised using the technique proposed for the general appliance model discussed above.

Finally, we realized most of the literature are focused on a new method or improving existing algorithm. They did not discuss the possible post-processing methods that would improve the results. We are proposing a K-NN based classification technique as a possible post-processing tool for NILM. We have discussed K-NN for energy disaggregation before in the supervised learning section.

The theory behind the algorithms and their details are presented in the algorithm development section.

2.4 Performance metrics

In terms of evaluating NILM algorithms, there are no benchmarks yet. Different metrics were used depending upon the aim, methods used. For example, certain studies focus only on detecting active appliances for which Root mean square error is not a suitable metric. Even there is literature exists that focuses on new metrics to evaluate NILM[134]. A comprehensive review of the metrics used in NILM can be found in [135]. However, in recent years, efforts have been taken to benchmark metrics and algorithms for comparison purposes. This is important for accelerating research and deploying NILM in the real world. The most used metrics in NILM research are shown in Table 6. The metrics used in this thesis are Precision, recall, F-1score. Our motive was only to find the on/off state of the appliances, so we considered the output as true positive and the appliance is on if both algorithm output and true value are above a small certain threshold say 10Wh. Similarly, we calculated True Negative, False Positive, False Negative to get the performance metrics.

Table 6: Most used performance metric in NILM

symbol	Performance metric	Equations
RMSE	Root-mean-square error	RMSE= $\sqrt{\frac{1}{T} \times \sum_{t=1}^{T} \lvert \bar{y}\,t - \bar{\hat{y}}t \rvert}$
MAE	Mean average error	MAE = 1/T $\times \sum_{t=1}^{T} \lvert \bar{y}_t - \bar{\hat{y}}_t \rvert$
D	Disaggregation percentage	D=$(\sum_{i=1}^{k} E_i)/E_{tot}$
TECA	Total energy correctly assigned	TECA=1-$\frac{(\sum_{t=1}^{T}\sum_{i=1}^{K} \lvert \hat{Y}t(i) - yt(i) \rvert)}{2\sum_{t=1}^{T} \bar{y}t}$
DE	Disaggregation error	DE=($\sum_{t=1}^{T}\sum_{i=1}^{K} \lvert \hat{Y}_t^{(i)} - y_t^{(i)} \rvert$)/2
A_{cc}	accuracy	A_{cc} = (TP + TN)/(TP+FP+TN+FN)
P	precision	P= TP/(TP+FP)
R	recall	R= TP/(TP+FN)
F1	F-measure	F1=(2 ×P ×R)/(P+R)

In Table 6 where $\bar{y}t$ is the original signal, $\bar{\hat{y}}t$ is the disaggregation signal, Etot is the total energy of the original signal, k is appliances signals number and T is sampling number; ŷ is separated signal,y is the original signal, TP is the true positive of working appliance, FP is the false positive of working appliance, TN is true negative and FN is the false negative.

2.5 Popular datasets

To avoid the complexity of time-consuming and costly data collection, academic researches are usually focused on already available datasets. There is a limited number of datasets available restricting the progress of the NILM. The most popular datasets are REDD [136], UK-DALE[137], BLUED[138], REFIT[139], PLAID[140], DRED[141],WHITED[142]. Some other datasets available to the public can be seen in [143]. In Table 7 different popular datasets and their resolution, amount of appliance types (classes), and quantity of appliance for each type (variety) for NILM are presented.

The dataset used in this thesis is a subset called Irise from the European project residential monitoring to reduce energy use and carbon emissions in Europe (REMODECE). The Irise dataset consists of energy consumption readings of 100 houses for a whole year in France. The sampling time is 10 minutes. It is important to test the algorithms for different datasets to validate the performance. this why we chose the Irise dataset. One can find the details about REMODECE in [144].

Table 7: Different popular datasets and their details

			Appliance		
Data set	**Bit**	**Fs**	**Classes**	**variety**	**Purpose**
REDD [136]	24	15 kHz	~ 20	10	house demand
BLUED [138]	16	12 kHz	~ 30	~ 1	house demand
UK DALE [137]	20	16 kHz	~ 40	~ 1 – 3	house demand
PLAID [140]	16	30 kHz	~ 12	~ 20	transients
HFED[145]	16	5 MHz	15	1	spectral traces
WHITED[142]	16	44 kHz	46	1 – 9	transients

Having presented the relevant research and datasets and performance metrics, we give a brief introduction to application types and the NILM working environment.

2.6 Appliances types

Appliances can be classified into three types based on the operational states and power consumption behaviour.

- On/off appliances
- Multi-state appliances
- Infinite state appliances

2.6.1 On/off appliances

This kind of appliance has only two states on state and off state. Appliances consume one specific amount of power all the time when active. Examples of this kind of appliance are toaster, light bulb, water heater. Mostly on/off appliances are purely resistive or with small reactive component [16]. This property of the on /off device can be used to detect the change in state and type of this kind of device.

2.6.2 Multi-state appliances

Multi-state appliances have more than one state of operation. Each state has specific power consumption. Multi-state appliances are also called finite state machines. An example of such a multi-state device is the Washing machine. When it is working, it involves multiple states of operations like washing, rinsing, drying and each state consumes different specific power.

2.6.3 Infinite state appliances

In this type, the number of observable states is not finite. The power consumption changes continuously without any clear observable change steps. Examples of this type of devices are laptop, TV, light dimmer. This type of device is difficult to model and detect. This type is also called continuously variable power appliances.

2.7 Working scenario

The performance of the NILM varies from each house due to many factors. It is estimated in the study [76] that typical households contain 30-50 appliances and their power consumption varies in a wide range(0-3000W). These appliances operate for a different duration(0-24h) and at a different time of day depending upon the occupant's preferences. A simple power demand analysis presented in the study [9] shows that majority of household energy demand is due to relatively few high energy demand devices like the water heater, dishwasher, washing machine, microwave oven, and few long duration low consumption devices like light bulbs(all bulbs considered), Wi-Fi router, Fridge. This is true and validated for the houses used in this work and results are presented in section 3. Considering these, NILM should tackle these factors and successfully find the high consumption device's demand to implement demand response in near future.

Having described the appliance types and working scenario we conclude the section with mentions of some applications NILM can offer in the next subsection.

2.8 Applications

We focused on the feasibility of NILM for demand responding in this thesis and described how it can offer it. EU's demand response market will worth 3.5 billion US dollars by 2025 [146]. Other than that, NILM can offer various services. Itemised energy bill to provide feedback to the customers. Customers can know about detailed energy consumption and can take appropriate decisions to save energy and cost. The American Council for Energy-Efficient Economy (ACEEE) [147] conducted an analysis that estimated the annual energy savings of 12% when provided real-time feedback to appliances level. In addition to this NILM can be used to provide preventive maintenance of electric appliances like Heating,Ventilation and Air Conditioning (HVAC) and finding malfunction which improves energy efficiency. For example, a faulty refrigerator can be replaced with an energy-efficient one [148]. For utility companies, this helps for proper planning, scheduling, and delivering power to the customers and it offers companies to better understand its customer and provide better services like a dynamic tariff. A home energy management system is another application proposed for NILM. NILM can provide smart home solutions with no or fewer sensors with its ability to find real-time appliance power consumption. The global market value of HEMS will worth 6 billion US dollars by 2025 [149]. in addition to this, there are other use cases proposed such as assisted living for elderlies, illegal load detection, targeted marketing, appliance product improvement, building performance analysis, and energy management. Although, it should be noted that different applications might need different technical requirements. For example, for demand response, NILM should be in real-time whereas for itemised bill it does not need to.

Having presented the relevant researches, performance metrics, dataset, and brief introduction into appliance types, working scenario, and applications of NILM, we move on to the work done in this thesis from the next section starting with exploratory data analysis.

3 Exploratory data analysis

Before going on into the disaggregation algorithm we present simple data analysis to present some insights on the train house's and test house's appliance consumption. These results will help to interpret the algorithm performance, shortcomings, and feature's importance. It should be noted that we did not use any of the test house's results in the algorithm. The algorithm is completely unsupervised for test houses. We are not also discussing the results in detail as this thesis motive is NILM and not data analysis. The results are only for interpreting the algorithm's performance and we also discussed how some of these parameters could be used in post-processing in the Future works section.

We would also like to make aware the readers that other than basic pre-processing no effort has been taken for missing values. This will not affect our NILM results because only the Time of use (TOU) and Day of the week (DOW) is used which will not change largely if there is missing data for a day or two even a week. Houses for training and testing are selected by examining appliances consumption and made sure that there are less/no missing values and appliances of the same type has different power rating and also diverse consumption patterns among occupants to have a real-world scenario.

we describe the parameters analysed in the next section before presenting results.

3.1 Parameters analysed

Day of usage
Certain appliances are used a few times a week. Day of usage describes how many times the appliance has been used on a particular day of the week(mon-sun).

Turned on month variation
Seasons and weather might influence appliance usage patterns for a certain appliance. Turned on month variation describes how many times the appliance has been used for each month.

Average Hourly consumption value
Hourly consumption value describes the average amount of power consumed by the appliance at a particular hour from the start and end date of the analysis.

Average day of the week consumption value
Day of the week consumption value describes the average amount of power consumed by the appliance on the particular day of the week(mon-sun) from the start and end date of the analysis.

Appliance contribution to total consumption
This value describes the ratio of appliance consumption to total consumption.

Total and average monthly consumption value
monthly consumption value describes the total and average amount of power consumed by the appliance at a particular month from the start and end date of the analysis.

Average operation time per usage
Operation time per usage describes the average operational period per usage of the appliance from the start and end date of the analysis.

Number of times turned on per week
This value provides how many times the appliance has been turned on per week on average.

Noise

We consider noise as other appliances consumption contributed to the consumption when the appliance of interest is operating.

3.2 Appliances of Interest

It's difficult for very low-frequency algorithms to find low consumption devices. so we decided not to focus on low consumption devices even though some might be the feasible appliance for demand response. Among high consumption devices, we chose five devices with the assumption that occupants might be willing to defer these appliance usages than willing to defer other appliances usages. The chosen appliances are a water heater, electric heating, washing machine, clothes drier, dishwasher.

3.2.1 Water heater

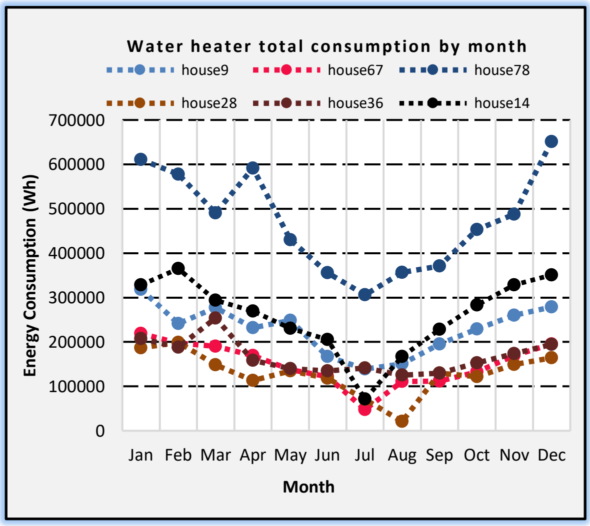

Figure 4: Water heater total consumption by month

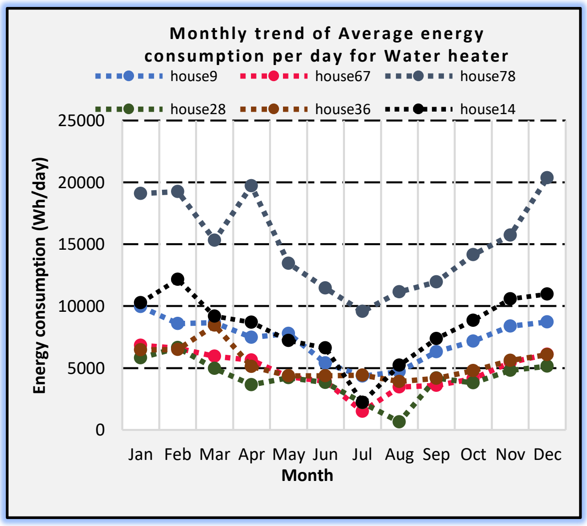

Figure 5: Monthly trend of Average energy consumption per day for Water heater

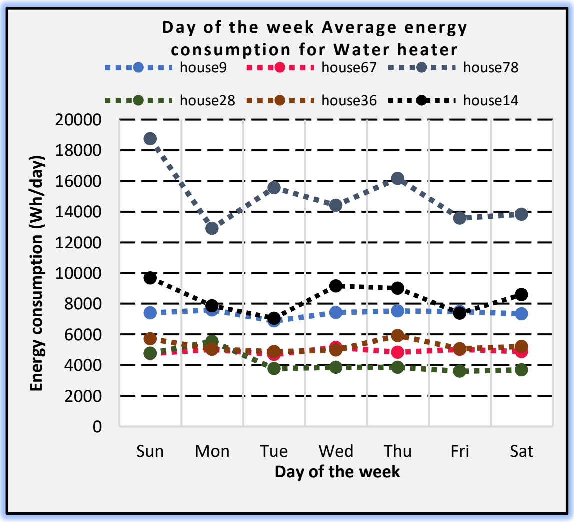

Figure 6: Day of the week Average energy consumption trend for water heater

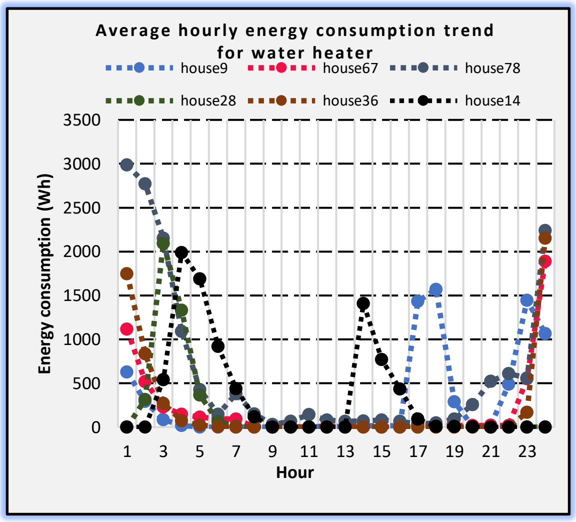

Figure 7: Average hourly energy consumption trend for water heater

One can observe seasonal, hourly variations, and the number of times turned on is more than once per day in some houses from Figures 4,7,8. There is no big day of the week variations observed in Figure 9. The water heater is one of the big contributors to the total consumption during summer except for house 78 due to unusual electric heating usage which can be seen in Figure 10. Table 8 indicates that During its operation, there is less noise which is advantageous to NILM. Operational hours are long.

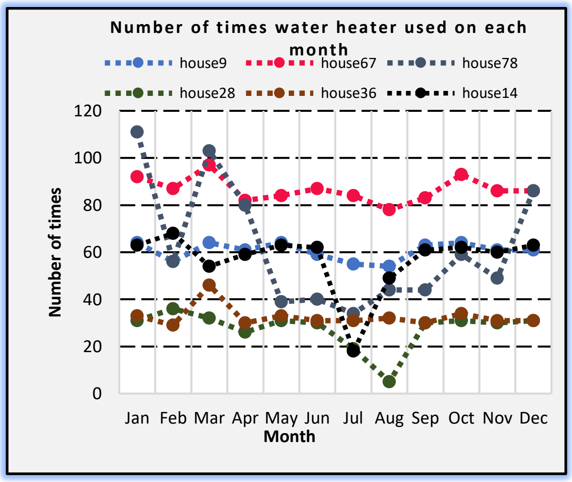

Figure 8: Number of times water heater used each month

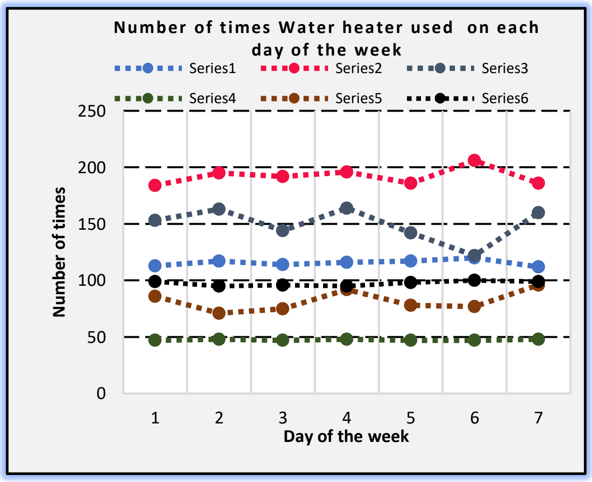

Figure 9: Number of times Water heater used on each day of the week

Table 8: Data analysis results of Water Heater

Water heater	Average operation time (minutes)	Noise (Percentage)	Number of times Used on per week
House 9	120	0.74	14
House 67	90	3.55	20
House 78	135	0.19	14.5
House 28	91	0.57	6.25
House 36	135	0.28	8
House 14	125	2	12.82

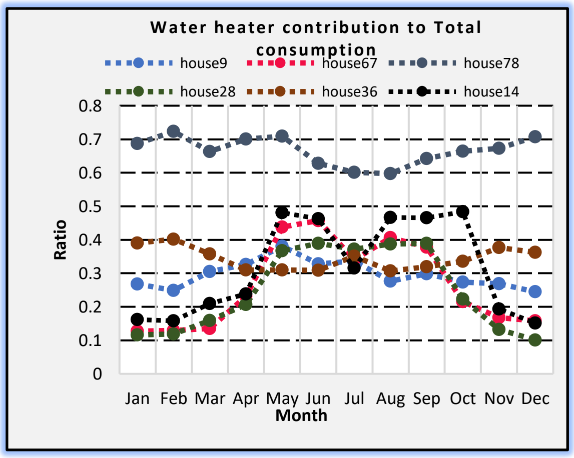

Figure 10: Water heater contribution to Total consumption

3.2.2 Electric Space heater (ESH)

Figure 11 shows that It's very unlikely this appliance is used during summer. In Figure 13, one could find an exception in house 67 where weekday consumption is more than weekend consumption. One could also find that there is no common trend among all houses in hourly variations in Figure 14. Some houses show little hourly trend variations. These two results indicate that electric heating is very much coupled with people's individual comfort and habits.

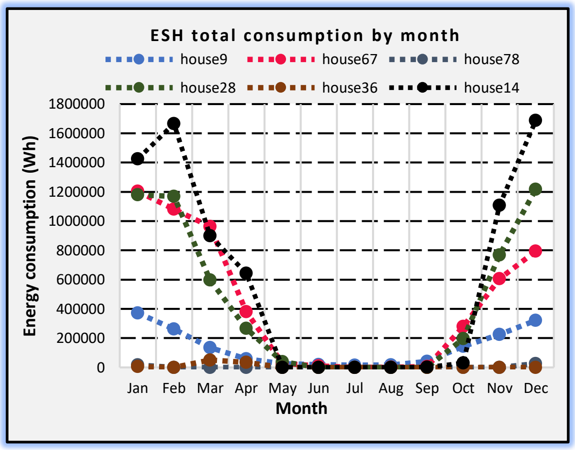

Figure 11: ESH total consumption by month

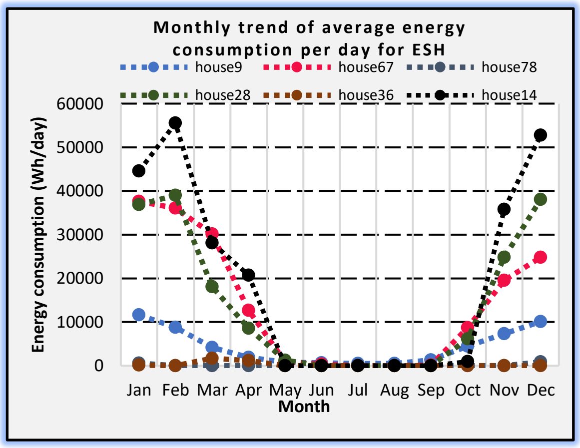

Figure 12: Average energy consumption/day trend over months for ESH

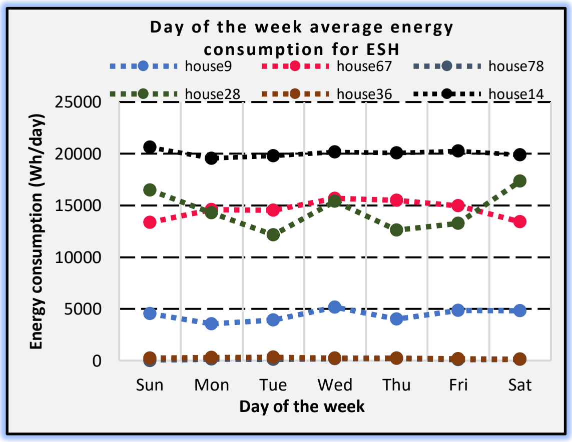

Figure 13: Day of the week average energy consumption trend for ESH

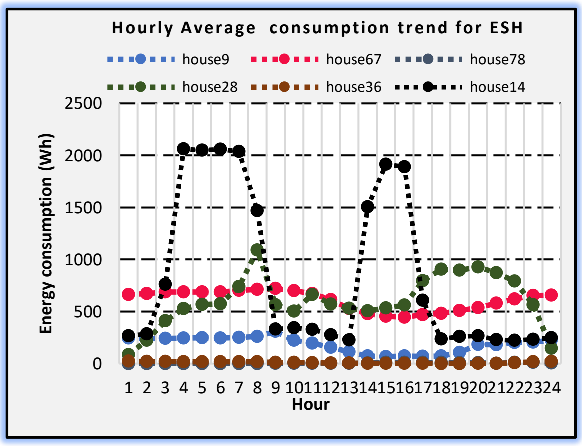

Figure 14: Hourly Average consumption trend for ESH

Figure 17 shows that The biggest contributor to total consumption in winter and Table 9 indicates that operational time is much longer. Like the water heater, the Noise percentage during its operation is less.

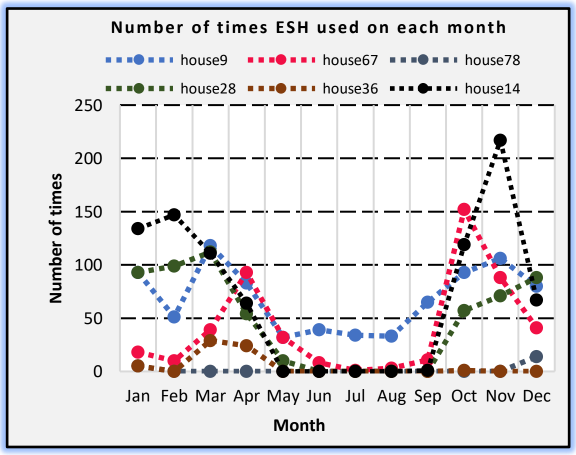

Figure 15: Number of times ESH used on each month

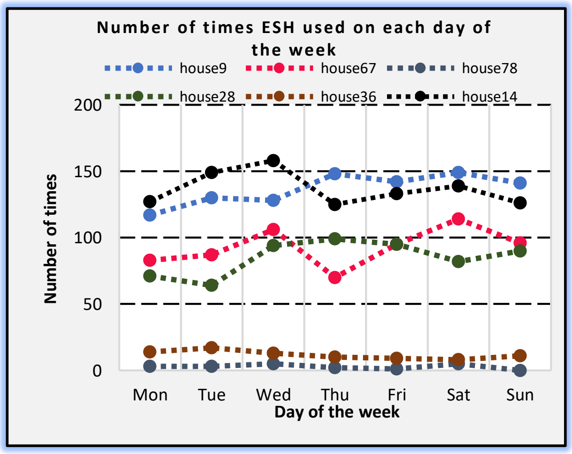

Figure 16: Number of times ESH used on each day of the week

Table 9: Data analysis results for Electric space heater

ESH	Average operation time (minutes)	Noise (Percentage)	Number of times used on per week
House 9	180	3.39	15.5
House 67	480	0.63	10
House 78	90	1.36	0.29
House 28	330	3.6	1.19
House 36	245	0.38	10.72
House 14	200	0.63	16

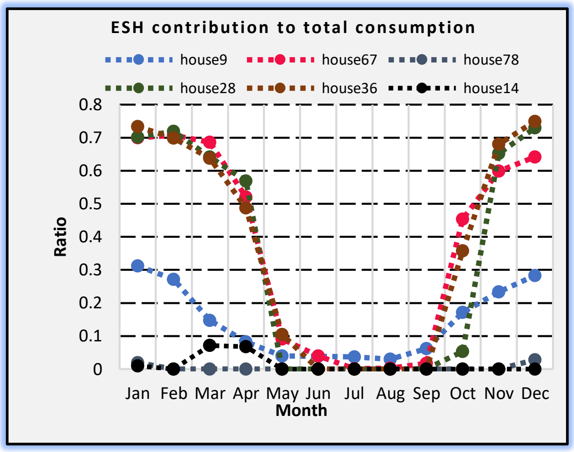

Figure 17: ESH contribution to total consumption

3.2.3 Washing Machine

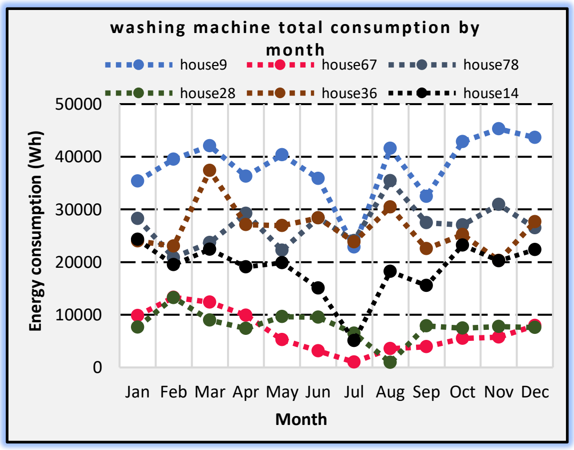

Figure 18: Washing machine total consumption by month

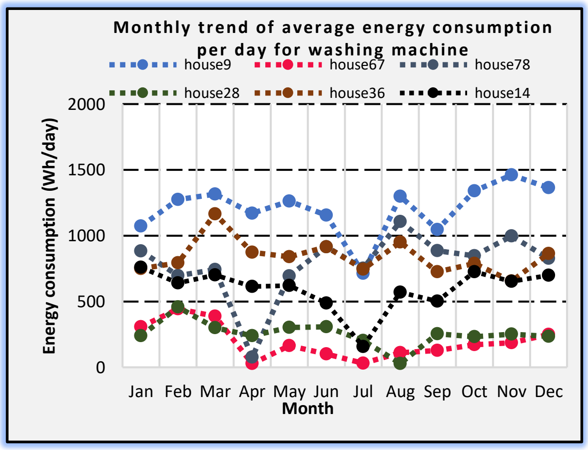

Figure 19: Average energy consumption/day trend over months for washing machine

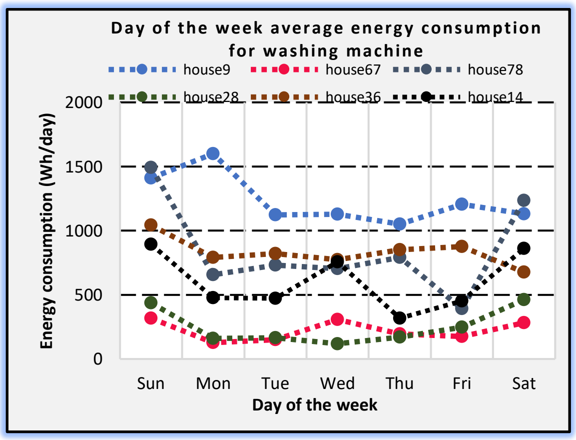

Figure 20: Day of the week average energy consumption

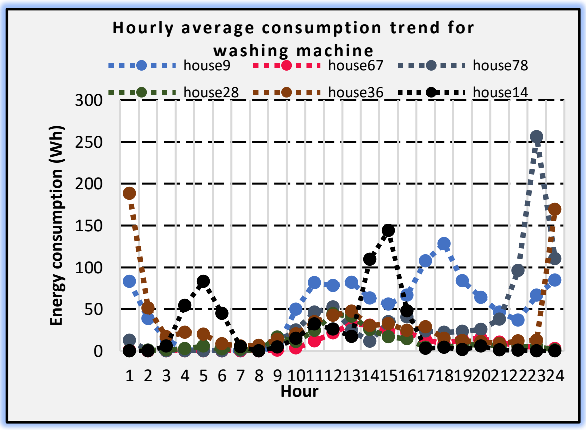

Figure 21:Hourly average consumption trend for washing machine

Figure 18 shows that there is less monthly trend variation and contrary to the water heater and electric heater, one can clearly see the day of the week variations from Figure 20. One interesting result is that for house 36 number of times turned-on on Monday is more than Sunday, but the average consumption is less than Sunday's which might indicate people might use quick wash during the weekday. Table 10 indicates that the noise percentage is higher which indicates trouble for NILM.

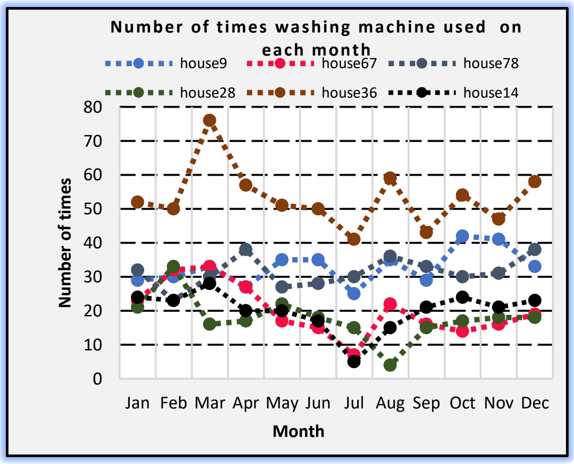

Figure 22: Number of times washing machine used on each month

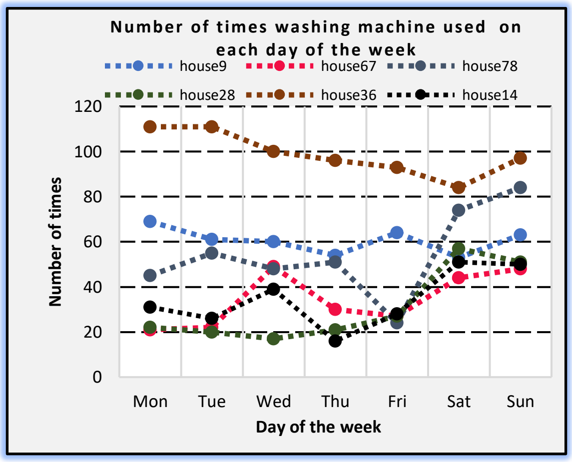

Figure 23: Number of times washing machine used on each day of the week

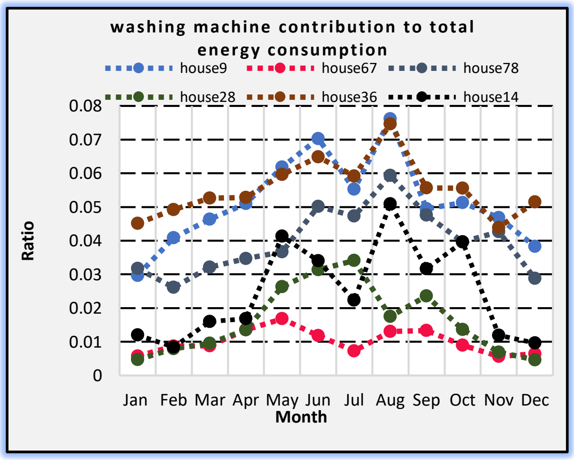

Figure 24: Appliance contribution to total energy consumption

Table 10: Data analysis results for Washing machine

Washing machine	Average operation time (minutes)	Noise (Percentage)	Number of times turned on per week
House 9	86	6.5	7.54
House 67	75.5	11	4.58
House 78	62.9	10.95	7.2
House 28	69.3	11	3.4
House 36	54.5	5	12.15
House 14	74	17.75	4.5

These results are a perfect example of why one should not develop a feature for NILM based on one simple analysis. For example, if one uses the average number of times turned-on per week which might not be equally distributed on all days but depend upon occupants' habits. Figure 24 shows that The contribution of the washing machine is very less to total consumption which would provide no real financial benefit to occupants involved in demand-side management. The average operational time is similar and lesser than the other two appliances discussed above.

3.2.4 Clothes dryer

Figure 25 shows that Unlike washing machine, clothes dryer shows seasonal variations. If one compares the hourly trend of washing machines and clothes dryer from Figures 21 and 28 for house78 one can see that clothes dryer were used in the morning but washing was not.

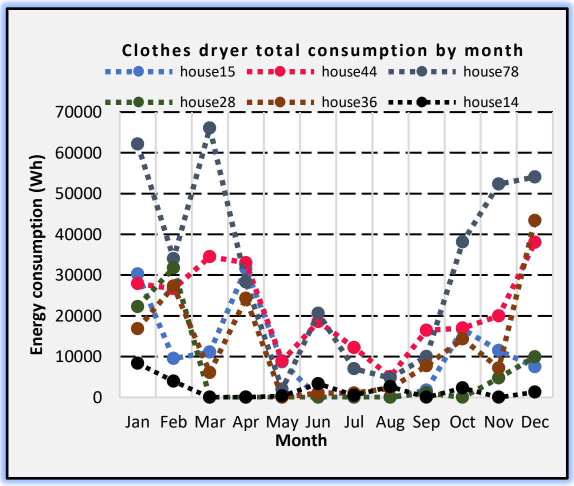

Figure 25: Clothes Dryer Total consumption by month

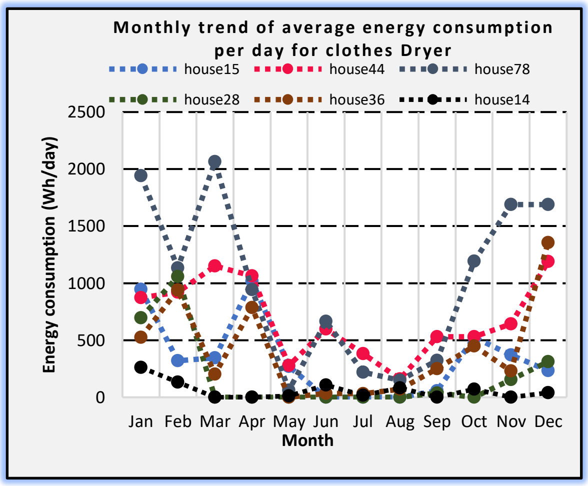

Figure 26: Average energy consumption/day trend over months for Clothes Dryer

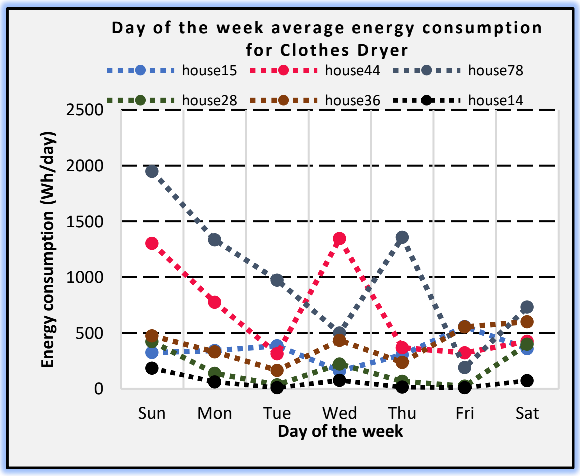

Figure 27: Day of the week average energy consumption for Clothes Dryer

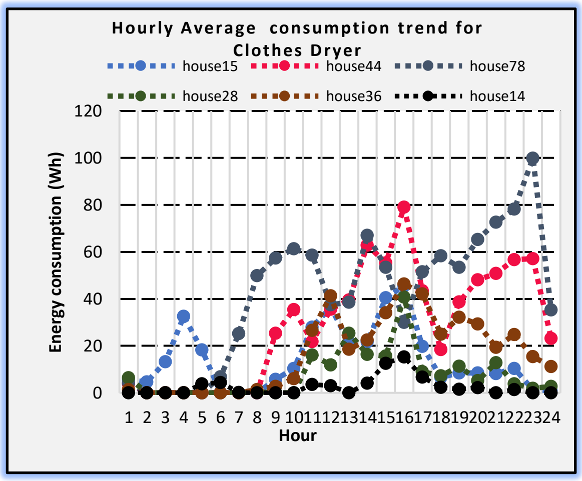

Figure 28: Hourly Average consumption trend for Clothes Dryer

Comparing the number of times turned on each month from Figures 22 and 29 also indicates clothes dryer was not always used after washing even during winters. This describes people's preferences and habits. So, a simple washing machine - clothes dryer correlation as a feature might not provide the desired result. Table 11 indicates All other results like noise, ratio, operational time are similar to the washing machine.

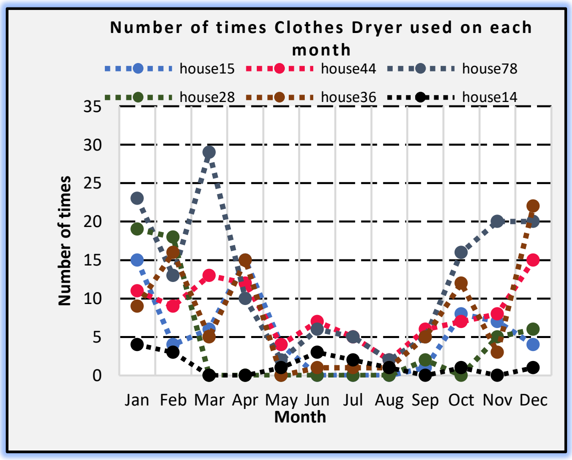

Figure 29: Number of times Clothes Dryer used on each month

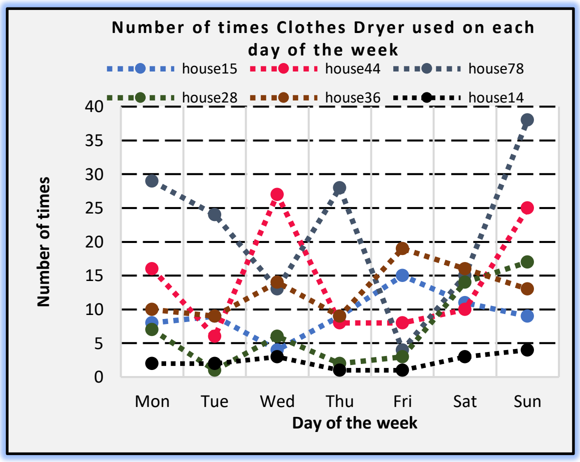

Figure 30: Number of times Clothes Dryer used on each day of the week

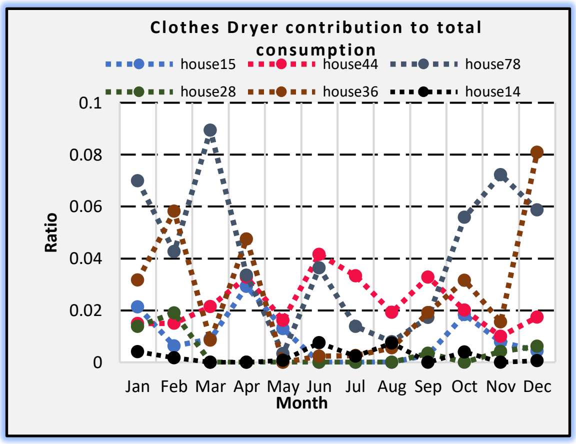

Figure 31: Clothes Dryer contribution to total consumption

Table 11: Data analysis results for Clothes Dryer

Clothes Dryer	Average operation time (minutes)	Noise (Percentage)	Number of times turned on per week
House 78	90	5.6	2.84
House 28	50	6.2	0.88
House 36	50	1.5	1.6
House 14	72.5	14.18	0.27
House 15	75	3.76	1.13
House 44	105	3.65	1.7

3.2.5 Dishwasher

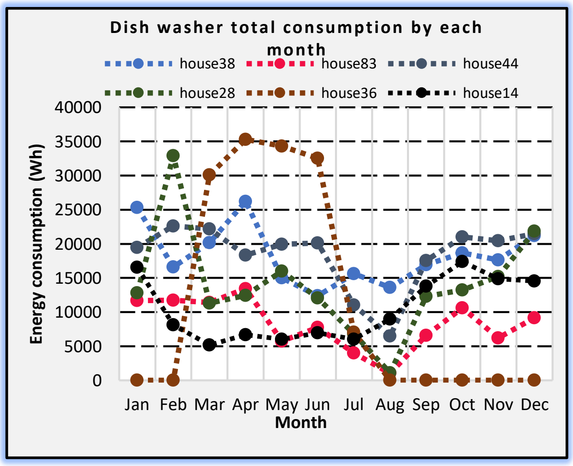

Figure 32: Dish washer total consumption by each month

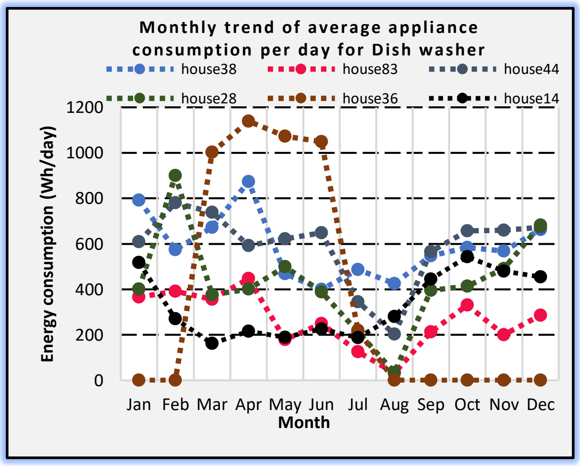

Figure 33: Average consumption/day trend over months for Dish washer

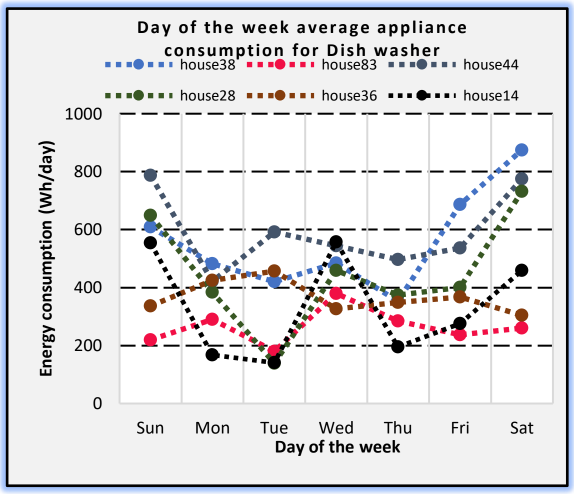

Figure 34: Day of the week average appliance consumption for Dish washer

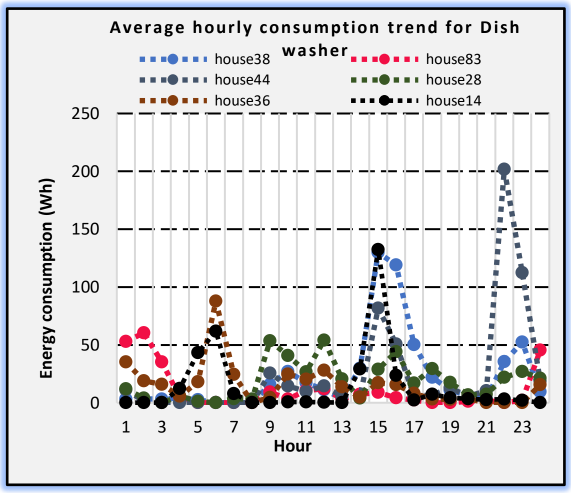

Figure 35: Average hourly consumption trend for Dish washer

From Figures 32 and 34, One could observe seasonal variation and day of the week variations. Unlike washing machines and clothes dryer, from Figure 35 one can see a common hourly trend among houses three peaks during food time and one either late night or early morning peak due to people's habit. This factor can be included in NILM. Table 12 indicates that other parameters like Noise, Ratio, operational time are similar to the washing machine, and clothes dryer.

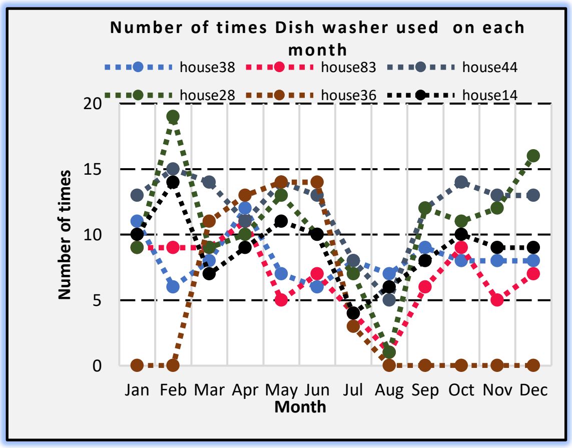

Figure 36: Number of times Dish washer used on each month

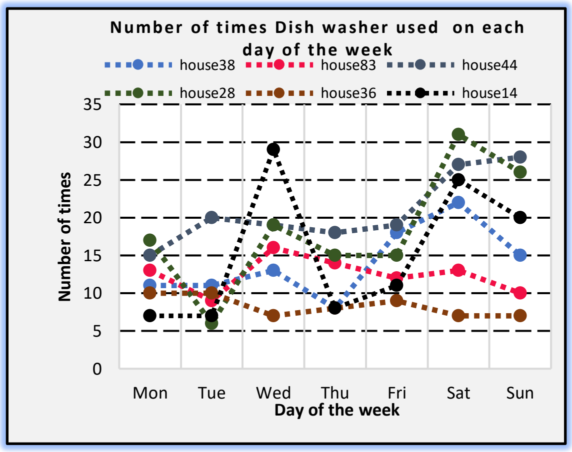

Figure 37: Number of times turned on each day of the week

Table 12: Data analysis results for Dishwasher

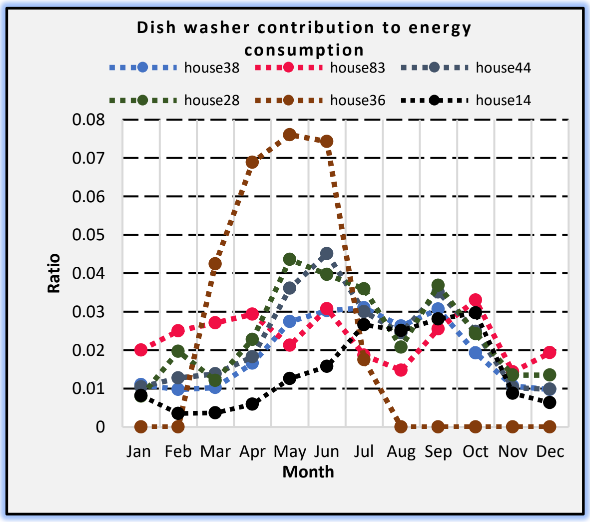

Figure 38: Dish washer contribution to energy consumption

Dish washer	Average operation time (minutes)	Noise (Percentage)	Number of times turned on per week
House 78	50	4.4	2.33
House 28	62	9.7	2.66
House 36	70	1.19	1
House 14	65	12	2
House 15	83	5.55	1.8
House 44	58	5.9	1.6

Having presented and discussed the data analysis results of important parameters for selected appliances , we move on to the details of the Methodologies employed in this thesis in the next chapter.

4 Methodology

This section explains the theory behind the Machine learning algorithms we used in this thesis for solving NILM. As we mentioned earlier, this thesis aims to develop methods that use the data from other houses to disaggregate consumption in unforeseen houses. The work done in this thesis can be considered to be focusing on three modules: data extraction module, model development module, post-processing module. The pictorial representation of which can be seen in Figure 39 below. Data extraction module and model development module are discussed individually for each method under its own topic. In this section, We start with the theory of the HMM model and how HMM with the appliance general prior model helps to identify the appliance's features. After that, we explain the tuning steps and disaggregation steps. Second, we explain the theory behind variational Auto-encoder and how it can be adopted for NILM. Third, we present the details of the temporal classification algorithm and how the supervised method is converted into an unsupervised method. Finally, we give the details of our contribution to the NILM: K-NN based post-processing.

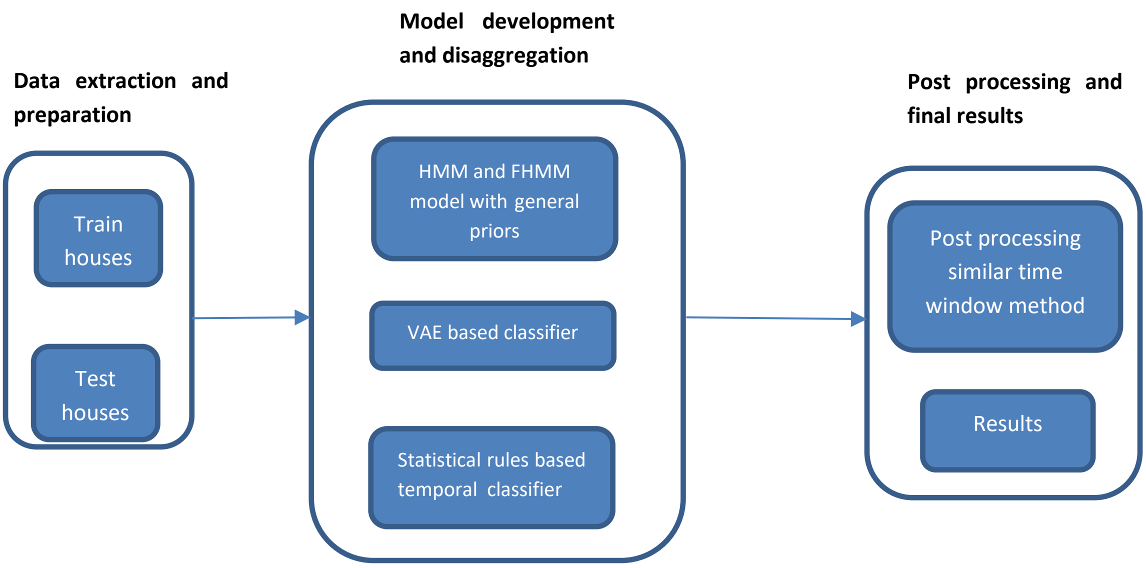

Figure 39: Modules focused in this thesis

4.1 Hidden Markov Model

HMM, based on Markov chains is suitable for probabilistic modelling of sequential data. Markov chain models the sequential data in which an observed variable is dependent only on the previous state but not on the other observed states.

HMM is a variant of the Markov chain model in which the discrete state is not directly observed but inferred through continuous variable it emits. A continuous variable is a probabilistic distribution function of the state. Hidden states are represented as $Z=(z_1, z_2,\ldots\ldots z_T)$ $z\epsilon(1,\ldots.K)$, where K is the number of discrete states that hidden state can take and T is the length of the sequence. The continuous observed variables are represented as $X=(x_1\ldots\ldots x_T)$ $x\epsilon\ \mathbb{R}$. In our case, the hidden state represents the appliance working state (on/off or ex: washing, rinsing), and observed values represent the power demand of that appliance state.

HMM can be defined with three parameters. First, the initial probability of the states that can be in sequence at t=1. It can be represented as vector π such that:

$$\pi=p(z_1=k) \qquad (1)$$

Second, the transition probabilities from state i at t-1 to state j at t can be represented with matrix A such that:

$$A_{i,j} = p(z_t=j|z_{t-1}=i) \qquad (2)$$

Both initial and transition probabilities follow the categorical distribution.

Third, emission probability function Φ for each state which provides the probability of the observed variable is emitted by a certain state at the time t. Generally, the Gaussian distribution function is used for this purpose. The emission probability is represented as such that:

$$xt|zt,\ \Phi \sim N(\mu_{zt},\tau_{zt}) \qquad (3)$$

where $\Phi=\{\mu_{zt},\tau_{zt}\}$ and μ_{zt} and τ_{zt} are mean and precision of a state Gaussian distribution.

The above equations can be combined to provide the joint likelihood of the HMM model. In the NILM context, this joint likelihood can give us the probability of the sequence due to the appliance of our interest.

$$p(x,z|\theta) = p(z_1|\pi).\ \prod_{t=2}^{T} p(z_t\ |z_{t-1}, A)\ .\prod_{t=1}^{T} p(x_t\ |z_t\ , \Phi) \qquad (4)$$

HMM model can be represented as θ= (π, A, Φ)

HMM involves two steps: parameter identification and inference. Parameter identification involves identifying θ given the sequence of training data. inference involves finding the most probable sequence of hidden states given the model parameters and the observed sequence.

The forward-backward algorithm also known as the Baum-Welch algorithm[150] is popular for learning parameters and the Viterbi algorithm[151] is popular for inference purposes. However, sometimes approximate methods are used for both learning and inference to reduce computational complexity. Since both of these algorithms are very much popular, we are not discussing the mathematics behind it.

Baum-Welch algorithm is a special case of the Expectation Maximisation algorithm. It is an iterative dynamic programming approach. It finds the parameters of the HMM by estimating the probability of the observation for the initial parameters. After that using that probability re-estimates the parameters to increase probability. This loop goes on until it converges to local maxima. The first step is called E-step or expectation step in which it computes the expected state occupancy count and expected state transition count. The next step is M-step or Maximisation step in which it estimates the transition parameters and emission parameters using these two counts.

Viterbi algorithm is popular dynamic programming that makes use of dynamic programming trellis. The observation sequence is processed from left to right. Each state of the model has its own trellis. Each cell of the trellis at t represents the maximum probability that the model is in a certain state at t after observing t-1 observations. The maximum probability is calculated recursively by taking the most probable path which leads to this cell. Since each step is calculated recursively, at the final step, we have the maximum probability of the sequence and the most likely hidden states by backtracking the cells that lead to the maximum probability and the final state. The computational complexity of the exact inference for HMM is $O(k^2T)$ where k is the number of states and T is the number of time steps.

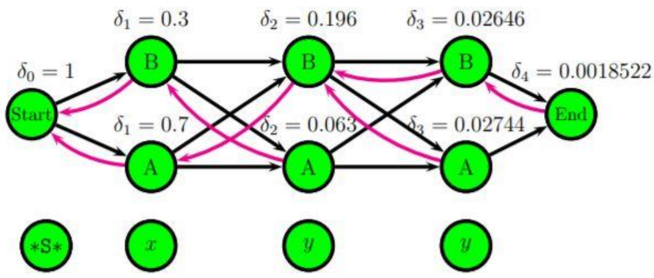

Figure 40: Example of a Viterbi trellis diagram [152]

An example of the Viterbi trellis diagram is shown in Figure 40 above. It starts with calculating the probability of A and B producing observable value x at t=1. After that, it calculates the probability of A producing y and transitioned from A in the previous step. Similarly, the probability for transitioning from B to A and producing y observable value is calculated. It compares the two probabilities and keeps the maximum one. similar steps are carried out for B state at t=2. Similar steps are carried out for the remaining time steps. In the end after having maximum probability and its source. It starts backtracking the source that resulted in maximum probability at the final step. In the above figure, the optimal states for producing the xyy sequence is ABB.

The steps involved in HMM with general priors for NILM is shown in the Figure 41. The steps involved are explained in the below subsections.

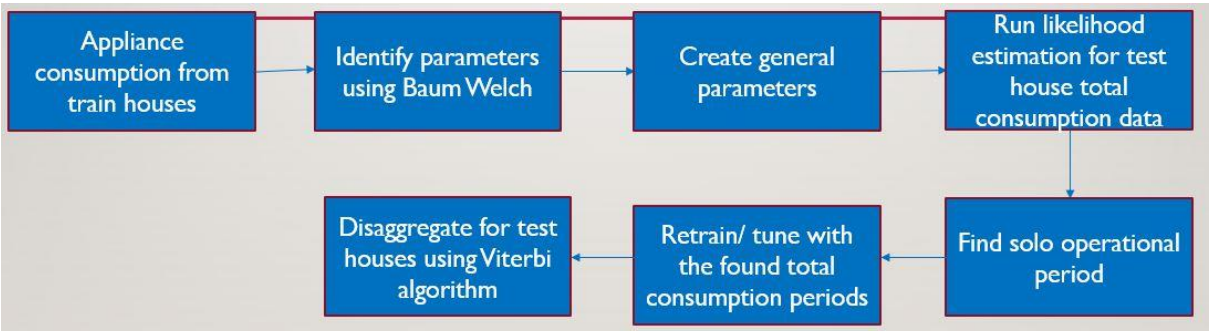

Figure 41: pipeline for NILM with HMM with general priors

4.1.1 Building appliance general model

In this section, we explained the procedure to build a general model for the appliance. we start by first modelling HMM individually for the appliance of interest for each of the train houses. We do that by first selecting the number of hidden states and prior values to each of the parameters and then run the training algorithm with appliance consumption data from the house. Usually in NILM number of hidden states is modelled as same as the physical operational states of the appliance. For example, a water heater is usually modelled with two hidden states as it has only two operational states (on, off). This method is suitable when the sampling frequency is less than 1 minute. When the sampling frequency is more than 5 minutes, there exists a reading between 0 and aggregate 5-minute reading as it is possible that the appliance was turned on/off between two sensor readings. when the frequency is less, it would not be a problem. Otherwise, the learning algorithm creates a wide Gaussian distribution to maximise expectations. The complexity increases for multi-state devices where two operational states can contribute equally to a single reading. So, for the water heater, we modelled HMM with three hidden states and for other appliances, we decided the states that are reasonable for all houses by exploring the histogram of data observed and running the training algorithm few times. Domain knowledge can be incorporated in prior values as it helps the training algorithm to find parameters faster when there is limited data. For example, the mean value for the Gaussian distribution function for off-state would be zero as the appliance will not consume any energy when it is off. We are using Gaussian distribution because it has been shown in the literature as a good fit for appliance power demand.

We used the Baum-welch algorithm to find the posterior probabilities of the appliance individually for each house. We used the pomegranate toolkit in python for parameter estimation. This toolkit is used because it is popular for HMM and flexible to use.

After finding individual model parameters for each house for a specific appliance, the next step is to create the general model. We represent general model parameters as $\theta_{general}$ such that:

$$\theta^{general} = (\pi^{general}, A^{general}, \Phi^{general}) \quad (5)$$

$\pi^{general}$ and $A^{general}$ also follow the categorical distribution and are created by drawing M samples from each individual posterior distribution obtained in the previous step and fitting the $\pi^{general}$ and $A^{general}$ to the samples. Similarly, $\Phi_{general}$ is also created. For the fitting, we used the NumPy library in python. General models for each appliance can be found in the appendix.

4.1.2 Tuning appliance general model

After creating the general model, the next step is to tune the general model to an individual house for the appliance using only the total consumption sequence. We do that by finding the windows of aggregate data in the total consumption sequence in which most likely only the appliance of interest was operating and training the HMM model again with the consumption data from that period. We find that window by running the likelihood estimation algorithm for the small overlapping windows of aggregate data and selecting the top 10 windows period with the most likelihood estimation. The likelihood estimation provides the probability that the sequence is generated by the HMM model. It is possible in that window of data other appliances are providing constant load. We solved that by subtracting the constant baseload in that window of data. the size of the window is determined by the maximum signature length encountered in the training data for the appliance. general model and tuned models are presented in the appendix.

The advantages of this method are we do not need sub-metered data from the house and less training time. This method has its disadvantages also. For example, not all appliances from all houses have a period in which it is a solely major contributor to total consumption. The type of noise signal when the appliance is operating will also make it difficult to find the period. Similar consumption patterns or multi-state devices with similar power demand will also affect this period identification. This problem can be overcome by incorporating domain knowledge like time of use, the period of use, day of the week. We did not use any domain knowledge in this thesis.

4.1.3 Disaggregation using Difference Hidden Markov Model (DHMM)

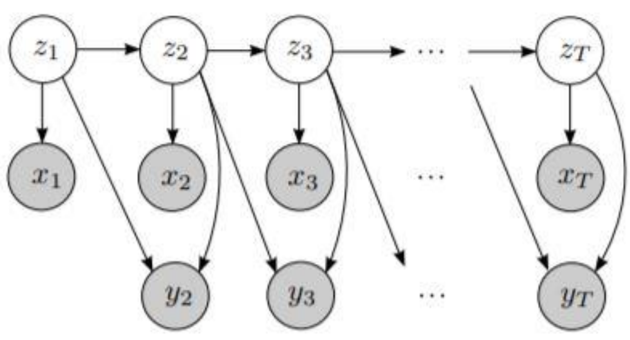

Figure 42:Difference Hidden Markov model variant [11]

Once the model is tuned it can be extended to a variant of DHMM by making difference in aggregate power between two consecutive steps as another observation sequence as shown in the Figure 42. In Figure 42, z represents the hidden state, x represents aggregate power demand, y represents the difference in aggregate consumption between steps.

we represent tuned model parameters as $\theta^{Tuned} = (\pi^{Tuned}, A^{Tuned}, \Phi^{Tuned})$. We are going to define the functions governed by these parameters below. These functions are required to do disaggregation.

The probability of an appliance's state at t=1 is given by

$$P(z_1 = k) = \pi_k \qquad (6)$$

The probability of an appliance changing from state i at t-1 to state j at t is given by

$$P(z_t = j \mid z_{t-1} = i) = A_{i,j} \qquad (7)$$

The gaussian distributed power demand of the appliance at a state is represented as:

$$P(w_{zt} \mid z_t, \phi) = N(\mu_{zt}, \sigma^2_{zt}) \qquad (8)$$

The probability that step change in aggregate consumption is due to appliance state change is given by the gaussian distributed function.

$$y_t \mid z_t, z_{t-1}, \phi \sim N(\mu_{zt} - \mu_{zt-1}, \sigma^2_{zt} + \sigma^2_{zt-1}) \qquad (9)$$

The above functions do not impose the constraint that the appliance can only be on and at a state only if the total consumption is above the mean consumption of the appliance state. In [ref], they imposed this constraint using erf function using total consumption reading defined as:

$$P(w_{zt} \leq x_t \mid z_t, \phi) = \int_{-\infty}^{xt} N(\mu_{zt}, \sigma^2_{zt})dw = 1/2\ [\ 1 + \mathrm{erf}\ (\ x_t - \mu_{zt} / \sigma_{zt} \sqrt{2}\ )\ ] \qquad (10)$$

This function makes the probability of the state tends towards zero if the total consumption reading is much below the mean consumption of the state and vice versa.

4.1.4 Disaggregation using extended Viterbi algorithm

Once we have the DHMM model with tuned parameters, any inference mechanism can be used for disaggregating appliance load from total load given only total load. We used the Viterbi algorithm as it is used by [9]. Viterbi algorithm is extended to process two observations and mean consumption constraints defined above. The Viterbi algorithm evaluates the joint probability of all the sequences in the model x, y, z using the equation given below:

$$P(x, y, z \mid \theta) = P(z_1 \mid \pi).\ \prod_{t=2}^{T} P(z_t \mid z_{t-1}, A).\ \prod_{t=1}^{T} P(w_{zt} \leq x_t \mid z_t, \phi).\ \prod_{t=2}^{T} P(y_t \mid z_t, z_{t-1}, \phi) \qquad (11)$$

The algorithm accepts the window for which joint probability is above a certain specific threshold. Appliance load consumption is assumed to be the mean consumption of the hidden states inferred for the windows accepted. In [9], the disaggregated data is subtracted from total consumption before disaggregating the next appliance to maintain the constraint that total consumption is equal to the sum of individual appliance consumption. In [106], the optimization approach is presented to overcome the disadvantage of the above method. We did not follow any of these two methods to know the algorithm performance. The results are presented and discussed in the results section.

4.2 Factorial Hidden Markov Model (FHMM)

FHMM is an extension of HMM in which there are multiple independent Markov chains. Each observation in the sequence is dependent upon multiple hidden states contrary to one hidden state in HMM. In the NILM context, FHMM is used to model multiple appliances into a single model. Each Markov chain represents one appliance. The graphical model of FHMM is shown in Figure 43.

Due to the conditional dependency of the multiple chains given an observed variable, the computational complexity for learning and inference is far higher than HMM. FHMM can be converted into equivalent K^N states HMM and exact inference can be done. However, the computational complexity would be $O(K^{2N}T)$. For larger N, the inference would be intractable. Due to this reason, often approximate methods are used as it provides a more tractable solution.

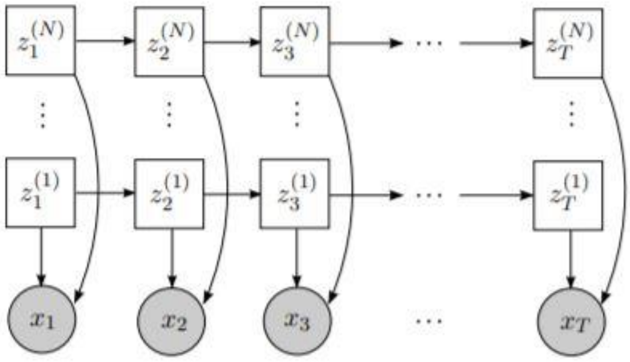

Figure 43: Factorial Hidden Markov Model [9]

Our motive for FHMM in this thesis is to explore the possibility of adapting the HMM method above to a set of appliances modelled using FHMM. The reason for this is that collecting submeter data is easier as less sensor is needed or if the dataset from the region only contains sub-metered data for a set of appliances. We could still get valuable information from sub-metered data. if the model is successful, this could potentially act as a virtual submeter. This virtual submeter could improve the individual HMM by providing information or feature. The well-known drawback of the above HMM method is that it can not perform well with noise or multistate medium-high consumption devices. NILM usually encounter multiple appliances operating together at the same time.

In the Irise dataset, for electric heating, there is no individual reading but single total electric heating consumption from multiple electric heaters is what available. We also do not know how many electric heaters are there. We developed and tested this FHMM for only Electric heaters. However, the method explored applies to other sets of appliances too.

In the work[153], the authors explored the possibility of using the Difference Factorial Hidden Markov Model (DFHMM) and Kronecker operation for disaggregating a set of Electric Space Heaters (ESH) from the total load. It was a supervised approach. In this thesis, we are using Additive-DFHMM for unsupervised modelling of a set of ESH and Gibbs sampling for inference. The methodology is explained in the following subsections.

4.2.1 FHMM general model

Like HMM general model, FHMM general model was created by selecting the number of chains, the number of hidden states, and prior values. As we did not know the number of heaters and their characteristics, these parameters are decided by exploring the training data and by testing different configurations with a parameter estimation algorithm. It was ensured that the final configuration would best represent the training data for all three train houses combined. The number of hidden states selected was three for each chain and the number of chains selected two. We found that the equivalent HMM model with 9 hidden states did not represent the data better than the FHMM model with 2 chains and 3 hidden states.

The initial probability of hidden states for our FHMM can be represented as π such that:

$$\pi = (\pi_m^1 \dots \Pi_m^N) \tag{12}$$

in the above equation, m represents the number of hidden states and N represents the number of chains. Each element in the π represents the initial probability of states in an individual chain and follows the categorical distribution.

The transition probability of hidden states for our FHMM can be represented as A such that:

$$A_{i,j} = (A_{i,j}^1 \dots A_{i,j}^N) \tag{13}$$

Here i, j represents the state at t-1 and t. N represents the number of chains. Each element in $A_{i,j}$ represents the transition probability matrix for individual chains and follows the categorical distribution.

The emission probability function for our FHMM observed variable can be represented as such that:

$$x_t | z_t^{(1:N)} \sim N\left(\sum_{i=1}^{N} \mu_{zt(i)}^{(i)}, \sum_{i=1}^{N} \tau_{zt(i)}^{(i)}\right) \tag{14}$$

we used Gaussian distribution to model each state's emissions and the summation operator in the above function indicates that the observed variable is the sum of the state emissions from each chain at a particular time.

The next step is to find parameters for individual houses. For learning we used the FHMM toolbox by Ghahramani in MATLAB is used. The toolbox has a forward-backward like an algorithm that implements exact E-steps. The M-step for FHMM is simple and tractable. However, E-step becomes intractable for a large number of hidden states and chains. The details of the algorithm can be found in appendix-B of [154]. since our model has fewer K and N, this algorithm can be implemented for our model. We used this toolbox because it is popular, simple to develop and implement models.

To create a general model, we followed the same steps we described in HMM general model section. For each chain, we created a general chain by drawing samples from corresponding chain parameters estimated for each house. Finally, we combine the general chains to create a general model.

Tuning the general model to an individual house also follows the same steps we described in the tuning general model section. Once the model is tuned, we can use ADFHMM for disaggregation which we explained in the below subsection. The general model and tuned model are shown in the appendix.

4.2.2 Additive Difference Factorial Hidden Markov Model (ADFHMM)

FHMM can be extended into ADFHMM by adding another observation variable that represents the difference in aggregate consumption between two consecutive steps. The graphical model of ADFHMM is shown in Figure 44.

The probability that step change in aggregate consumption due to state changes in the chains is given by the Gaussian distribution function

$$y_t | z_t^{(1:N)}, z_{t-1}^{(1:N)} \sim N\left(\left(\sum_{i=1}^{N} \mu_{zt(i)}^{(i)} - \sum_{i=1}^{N} \mu_{zt-1(i)}^{(i)}\right), \left(\sum_{i=1}^{N} \tau_{zt(i)}^{(i)} + \sum_{i=1}^{N} \tau_{zt-1(i)}^{(i)}\right)\right) \tag{15}$$

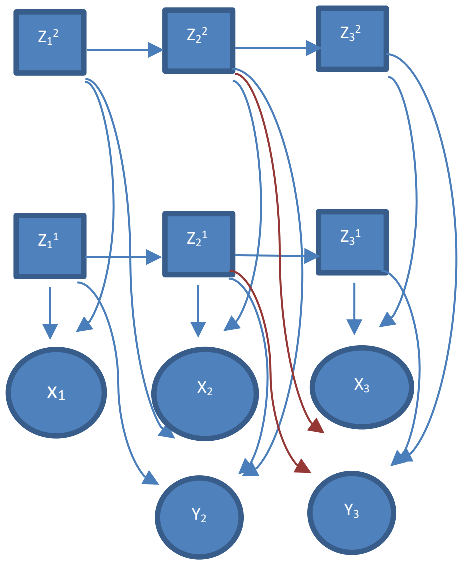

Figure 44: Additive Differential Factorial Hidden Markov Model

Since the exact inference for ADFHMM is computationally intensive. We used Gibbs sampling for inference to disaggregate ESH consumption from total consumption reading. We explained Gibbs sampling and disaggregation in the following subsection. The joint probability of all sequences in the model z^1,z^2,x,y using the equation below.

$$p(x, y, z^{(1:N)}|\theta) = \prod_{i=1}^{N} p(z_1^{(i)}|\pi^{(i)}) . \prod_{i=1}^{N}\prod_{t=2}^{T} p(z_t^{(i)}|z_{t-1}^{(i)}, A^{(i)}) . \prod_{t=1}^{T} p(w_{zt} \leq x_t|z_t^{(1:N)}, \phi) . \prod_{t=2}^{T} p(y_t|z_t^{(1:N)}, z_{t-1}^{(1:N)}, \phi) \quad (16)$$

The term $P(w_{zt} \leq x_t|z_t^{(1:N)}, \phi)$ is the erf function similar to the one used in the HMM model to impose the constraint that certain state combinations are not possible unless the aggregate consumption is above the sum of mean values of the states of each chain.

4.2.3 Gibbs sampling

Gibbs sampling is a Markov Chain Monte Carlo algorithm that can be used to approximate samples drawn from a multivariate distribution[155]. The idea of Gibbs sampling is that it repeatedly samples a value for latent variable conditional on all other variables in the model. it means for finding Hidden states of the sequence, it starts with random initialization of states for all time steps in the sequence and evaluates states probability for the first time step by keeping all other states of other time steps constant and updates the state with high probability state for the conditions of the model. it moves to the next time step and does the same by keeping the remaining states constant and updates the current state. These steps will go on until the algorithm converge. Gibbs sampling is simple to implement but slow since it depends upon the initial values for the states. Once Gibbs sampler finds the hidden states, we can easily find the joint probability of the sequences. The Pseudocode of Gibbs sampler is shown in Figure 45.

Based on the appliance-specific threshold, the algorithm accepts a window of data with a probability higher than the specified threshold. The disadvantage of this method is that we cannot know individual appliance consumption. Other disadvantages are it might not be suitable for a set of low consuming devices as the model can possibly confuse with single high consuming multi-state devices, the individual device with a similar pattern will also affect the disaggregation. However, our results for electric heaters are reasonable and it showed it can act as a virtual submeter for a set of appliances at least for space heaters. With some domain knowledge, it might be possible to implement this for other sets of appliances. The results are shown and discussed in the results section.

Algorithm 1 Gibbs sampler

Initialize $x^{(0)} \sim q(x)$

for iteration $i = 1, 2, \ldots$ **do**

$x_1^{(i)} \sim p(X_1 = x_1 | X_2 = x_2^{(i-1)}, X_3 = x_3^{(i-1)}, \ldots, X_D = x_D^{(i-1)})$

$x_2^{(i)} \sim p(X_2 = x_2 | X_1 = x_1^{(i)}, X_3 = x_3^{(i-1)}, \ldots, X_D = x_D^{(i-1)})$

$\vdots$

$x_D^{(i)} \sim p(X_D = x_D | X_1 = x_1^{(i)}, X_2 = x_2^{(i)}, \ldots, X_D = x_{D-1}^{(i)})$

end for

Figure 45: Pseudocode of Gibbs sampler [9]

The advantages of this method are that it requires less training time and data, it could act as a virtual submeter, and can provide additional information for the HMM model. The disadvantages are it is computationally very expensive, it can not specifically give individual appliance consumption. It is also prone to noise during tuning and disaggregation.

4.3 Variational autoencoder

An autoencoder is a Deep Neural Network (DNN) model that is trained to produce an output closer to the input with the minimum loss[156][157]. the unique feature of the autoencoder model is the diabolo shape. the main purpose of the autoencoder is to compress the data efficiently by finding the key features. The compressed latent representation of the input data is in the latent space at the bottleneck layer of the model architecture. The left side of the bottleneck layer is called the encoder also known as the recognition model because it recognizes the key features of the data and compresses it. the right side of the bottleneck is called decoder also known as the generative model because it generates the output based on low-dimensional latent points. In recent years, there is a huge interest in deep generative models[158][159]. The decoder model of the autoencoder can be used to create new content once it is trained. However, for the new content to be meaningful, the latent space should be regularised. It is difficult to ensure regularisation in the latent space as it depends on the distribution of the input data, the dimension of the latent space, and encoder architecture.

Regularisation can be included in the training process which leads to AE variant Variational autoencoder (VAE). VAE was initially proposed in [160]. in VAE encoder outputs are not deterministic as it is in VE. Instead, in VAE the encoder outputs are probabilistic distributions. Any probabilistic distribution can be used depending upon the input data. we used Gaussian distribution to represent input data in latent space. The reason the latent space is represented by distribution instead of the single point is that it provides an easier way to enforce regularisation. The principle is that input data is encoded as distribution in latent space and a point is sampled from the distribution. After that decoder uses this sample point and will try to reconstruct the input vector. VAE is trained by reducing the loss function. The loss function for VAE contains two terms, reconstruction term which reduces the error difference between the input vector and decoder output, and regularisation term which enforces the regularisation in latent space by forcing the encoder output closer to the standard normal distribution. The regularisation term uses Kullback-Leibler divergence to do this. The intuition here is that as encoder mean is forced to closer to 0, it prevents the encoder output distributions to be far apart in latent space and as the encoder covariance matrix is forced to be closer to identity, it encourages overlaps in the returned distribution. The mathematical theory behind VAE is outside the scope of this thesis. Details can be found in[161]. VAE can be used for various purposes like source separation, feature extraction, new content generation. Multiple variants of VAE, training methods are proposed depending upon the application. We are not discussing that as it is also outside the scope of this thesis. We use simple VAE for NILM. We explained the methodology in the below subsection. The graphical representation of VAE is shown in Figure 46.

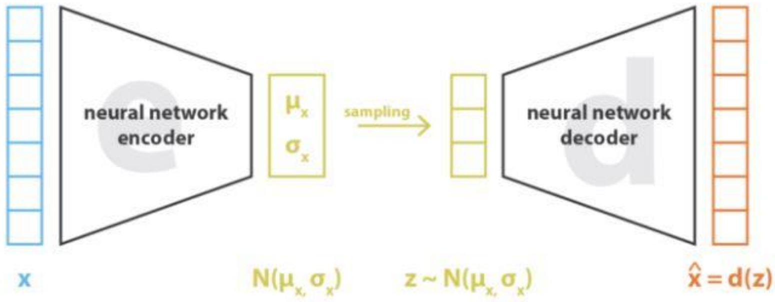

Figure 46: Variational autoencoder [162]

The Kullback-Leibler divergence between two probability distributions p and q can be given by the equation below.

$$D_{KL}(p||q) = \sum_{i=1}^{N} p(xi) \cdot (\log p(xi) - \log q(xi)) \quad (17)$$

The loss function used for VAE training can be represented as below.

$$\text{Loss}=|| x-\hat{x}||^2 + \text{KL } [N(\mu_x , \sigma_x), N(0,I)] \quad (18)$$

4.3.1 VAE based classifier for NILM

The principle used in this thesis for load classification is to extract features using a VAE encoder and classify them using Neural Network (NN) based classifier. At first, VAE is trained with the appliance data from train houses. Once VAE is trained, by taking only the encoder part and fixing the weight, it can be used as a feature extractor. Here the features are mean and variation of the distributions in latent space. Since we know the input data from which appliance, by creating the labels for features, NN based classifier is trained to classify loads in a supervised manner. Any Classifier algorithm can be trained on top of VAE. Once the training is finished, during testing, we used an encoder to extract features and a classifier to classify the sequence for the test houses.

For each appliance, individual VAE and NN classifiers are trained. The training data is prepared with a certain overlapping window of appliance data. the window length was decided based on the data encountered in the training set. The overlapping window was decided by multiple testing with different window lengths for each appliance and select the one that provides minimum loss function value. In the training dataset, consumption data of the appliance when it is off is not included. The training dataset is split randomly into train data and test data with a percentage of 70 and 30. Since the same appliance from the different houses has different consumption ranges, the data is normalized to bring the training data to a common scale.

For the VAE of each appliance, network architecture, number of layers, number of neurons, type of neurons, number of distributions for latent space were decided by testing multiple configurations along with multiple configurations for classifier and selecting the one that provides minimum loss function value and higher classifier accuracy. Classifier also included in this because sometimes the best configuration that provides minimum loss function might not provide unique features for classifier training. The optimisation algorithm to select these parameters was discouraged by the immense amount of time it needs. We considered Long-Short Term Memory (LSTM), Recurrent Neural Network (RNN), and Convolutional Neural Network (CNN) for the type of layers. LSTM and RNN are popular for their ability to model sequential data and CNN is popular for pattern recognition ability. the final parameters selected for VAE and classifier of each appliance are shown in the appendix.

The neural network is inspired by biological neurons. A neural network consists of a set of neurons commonly referred to as neurons and a set of directed edges between them. Weight is associated with each edge between two nodes. The value computed at each neuron is the weighted sum of the inputs. Usually, to provide flexibility, this value is added to the weighted bias. So, the value computed at each neuron can be represented as below

$$\hat{y} = \sum_{k=0}^{n} w_k . x_k + b \tag{19}$$

where W_k is the weight associated with input x_k and b is the bias. Activation function may be associated with each neuron. The type of activation function depends upon the purpose. For example, sigmoid activation can be used for probability output. Details of various activation functions can be found in [163]. So, the final value computed at the neuron with an activation function can be represented as below.

$$\hat{y} = f\left(\sum_{k=0}^{n} w_k . x_k + b\right) \tag{20}$$

where f () is an activation function.

The most popular algorithm to train neural networks is backpropagation introduced in 1985[164]. it uses the chain rule to find the derivative of the loss function with respect to all parameters in the model and adjust the weights by gradient descent. The loss function compares the predicted value with the actual value and provides how much difference is there. During training, the algorithm goes throw the samples multiple times randomly and computes the loss function, and adjusts the weights accordingly to minimize the loss function. Since the loss surface is nonconvex, there is no guaranteed training that reaches global minima. Details of various training algorithms can be found in [165].

Neural network architecture usually has multiple layers- an input layer, an output layer, and many hidden layers. Each layer has many numbers of neurons. Feedforward neural network (FNN) is a type of neural network architecture in which values propagate in a single direction from the input layer to the output layer. an example of FNN is shown in Figure 47.

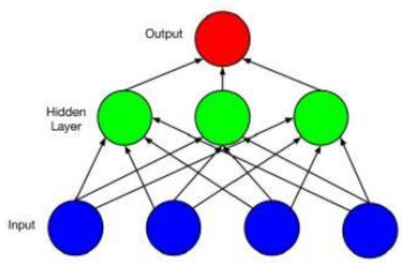

Figure 47: Feedforward Neural Network[165]

FNN does not have the notion of time. A recurrent neural network is a special type of neural network which allows the notion of time. It does so by introducing recurrent edges that span adjacent time steps. A simple RNN is shown in Figure 48. The edge with the pink line is the recurrent edge which provides neuron value to previous inputs along with current input. So, the value of neurons not only depends upon current input but also previous input. This is the reason for its popularity to model sequential data. details of various RNN and training procedures can be found in [165]. vanishing gradient is one of the well-known problems in training RNN. Long short term memory was introduced to overcome that. It is similar to the recurrent neural network except in hidden layers, the ordinary node is replaced with a memory cell. This memory cell ensures gradient can pass on multiple time steps without the problem of vanishing gradients. A good review of LSTM can be found in [166]

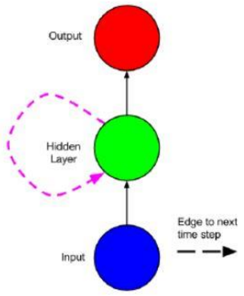

Figure 48: Recurrent Neural Network [165]**.**

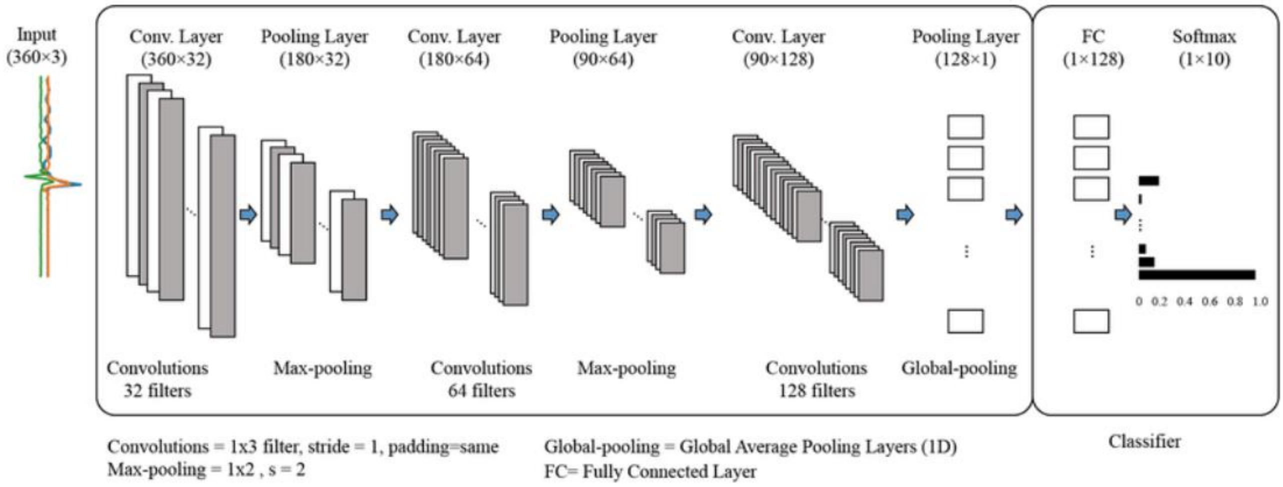

Figure 49: Multi-class classifier using Convolutional Neural Network [167]

A convolutional neural network is a type of neural network which contains three types of layers – convolutional layer, pooling layer, fully connected layer. the convolutional layer acts as a filter and extracts features from the input data and creates the feature map. The pooling layer is used to reduce the dimensionality of the feature map. The data is converted into output through a fully connected layer. when CNN is trained in a supervised manner, convolutional layer weights are adjusted to extract the right features to provide accurate output. Details of CNN can be found in [168][169].

We used CNN for NN based classifier due to its recent popularity in pattern recognition, object detection, classification, and other applications. The architecture selection is explained above. The final layer of the CNN classifier is the discriminative layer. This layer provides the probability of the class for the given input. The number of neurons in the discriminative layer depends upon the number of classes. In our case, there are only two classes "from the appliance of interest" and "not from the appliance of interest". One neuron is enough to provide the probability of the classes. An example of a CNN based Multi-class classifier is shown in Figure 49.

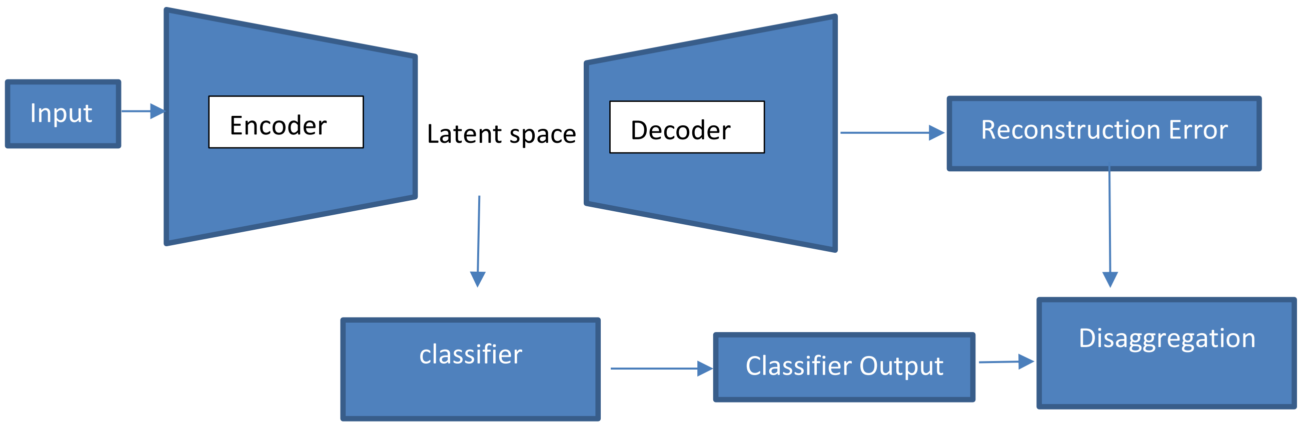

Figure 50: Variational Auto encoder based NILM

The steps involved in the VAE based NILM is shown in Figure 50. The algorithm accepts the window if it satisfies the following conditions: the probability output of the classifier is above a certain specific value and the reconstruction error of the VAE is less than a certain specific value. These specific values are decided by testing the model with training data and select the one that provides maximum accuracy. It should be noted that for the selected value, the model might have low accuracy for test houses than train houses. This is because the model is trained with train house data. for test houses, the entire process is unsupervised. The reason that reconstruction error is included for evaluating the window is that VAE performs poorly in reconstructing the input data if it has not seen similar data in training. Of Course, similar data from different appliances might affect the performance but it is a common disadvantage faced by all NILM algorithms. The inclusion of domain knowledge might help solve this problem. We used the python deep learning library to develop the models.

The advantages of this method are feature extraction is automatic, which can be used to generate synthetic data and can perform in real-time. The disadvantages are it requires a long training time, a large amount of data for training, training data should be general enough for the appliance, can not be tuned to a specific house.

4.4 Temporal classifier based on statistical rules

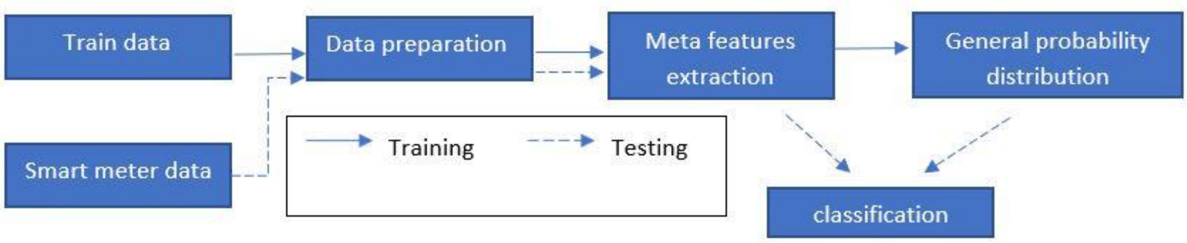

Figure 51: Methodology of Statistical rules based NILM

The third method explored in the thesis is based on representing the problem in such a way that can be understood by the propositional concept learners[170], [171]. This method involves attribute construction by knowledge extraction of the substructures in the training instance. By providing additional attributes, propositional learners can perform better than without these attributes. These substructures are called meta-features. Indeed, the choice meta-features depend upon the problem and application. In our case, meta-features are based on time and consumption values. The techniques of meta-features generation in the temporal domain[90] are employed to create the attributes. The meta-features used are defined in the meta-features subsection below.

The steps involved in this method for load classification is as follows:

1. Subsequences are generated using a temporal sliding window from the training data. the window length and sliding width are decided based on the data encountered in the training dataset. Data is normalized. Details can be found in the Appendix.
2. Meta-features are generated for the subsequences.
3. Generic meta-features are developed for each appliance. this step involves generalizing the meta-features obtained for the specific appliance from different train houses so that it can represent all appliances of a specific type. This step is similar to the HMM generic appliance model creation. We introduced this step to make the supervised method unsupervised for test houses. This step is explained in detail in generic meta-features creation.
4. Using generic meta-features, instances of test houses are classified. This step is explained in detail in the disaggregation subsection.

The steps involved in the Statistical rules-based NILM is shown in Figure 51. The Solid arrow in the figure indicates the training phase while the other indicates the classification.

4.4.1 Meta-features

The meta-features used in this thesis are introduced in this subsection.

Time of use

As the name suggests this feature will represent which time-period appliance has been used. 24hours is split into 24 periods with 1-hour intervals. Knowledge extractor counts the number of readings (appliance on) in each time period in the event window. We used the number of readings instead of just the time-stamp at which the appliance has been turned on because the appliance may have been turned on just before the next hour and the turned-on time stamp would mislead the information. The time of use can be represented as below.

$$TOU(h)=N;\ N\epsilon\ R,\ h\epsilon(1,24) \tag{21}$$

Where N represents the number of above zero value readings. h represents the hour

Day of the week

This feature represents on which day of the week the appliance has been used. Each day of the week is represented with numerical values as Monday being 0 to Sunday being 6. Day of the week can be represented as below:

$$DOWn= D;\ D\epsilon\ (0,6) \tag{22}$$

Max and min values positions

This feature represents the positions of the max and min values in the event window. If there is two or more max or min values. This feature will provide their positions also. It can be represented as below.

$$Pos_{max} \longrightarrow E(pos_{max})=max\ (E_i);\ i\epsilon(1,L) \tag{23}$$

$$Pos_{min} \longrightarrow E(pos_{min})=min\ (E_i);\ i\epsilon(1,L) \tag{24}$$

Where E_i represents energy value at the i^{th} position in the window, i represents the position in the window, L represents the length of the window.

Energy difference

This feature represents the energy difference of each step with respect to the certain reference in the instance. For example, if an instance contains 15 readings and if the 8^{th} position is the reference. this feature will provide details of how much energy difference is there in each step with respect to the 8^{th} position value. It can be represented as below.

$$E_{diff}(p) = E(p_{ref}) - E(p);\ p \in [0, L] \tag{25}$$

Where p represents the position in the window, L represents the window length, E represents the energy value at a position.

Gradient, Gradient ratio, Laplacian

Gradient ∇ represents a change in energy consumption between the current timestep and the previous time step. The gradient is calculated for all time steps in the window except first. Laplacian Δ represents a change in the gradient between the current timestep and the previous time step. Gradient ratio ∇_r represents the ratio between the gradient and the energy value at any given time step. These features provide trends in energy consumption in the event window.

$$\nabla(p) = E(p) - E(p - 1) \tag{26}$$

$$\Delta(p) = \nabla(p) - \nabla(p - 1) \tag{27}$$

$$\nabla_r(p) = \nabla(p)/E(p)\ ; \qquad p \in [0, L] \tag{28}$$

It is possible to include more meta-features like energy variation between timesteps, distances between current timestep to local maxima and local minima, mean, and standard deviation of the data in the event window. We left these features for future study.

4.4.2 Generic meta-features and probability distribution

This subsection explains how the features extracted from training data are used to construct generic value and probability distribution for these meta-features that can represent any appliance of a specific type. The generic value obtained is shown in the Appendix for each appliance.

Time of use

Time of use is a discrete variable. The data from each event window is added to create an overall value for TOU. It is possible to create categorical distribution to each house with the knowledge extracted. Once it has been extracted, it is possible to create a generic categorical distribution for TOU by sampling from each distribution and fitting the categorical distribution to the samples. Individual distribution is needed here to avoid the domination of data from the house in which the appliance has been used much higher than other houses.

Day of the week

Day of the week is also a discrete variable and the same procedure for TOU is employed for DOW as well to create the general categorical distribution.

Max and min values positions

This is also a discrete variable and the same procedure is used.

Energy difference, Gradient, Gradient ratio, Laplacian

These features are continuous variables. For each house, a general model needs to be created that represents all event window. This is achieved by constructing Gaussian distribution for all the variables of these features using the knowledge extracted. For example, the knowledge extractor extracts the energy difference between the current time step and the reference time step in the event window. If the event window contains 15 timesteps then there are 14 variables for the energy difference meta-feature. We constructed Gaussian distribution for each variable using the data from all event windows. After creating distributions for each house, general distribution can be created similarly by drawing an equal number of samples from each distribution and fitting the model to samples.

Energy disaggregation

Once general distributions are created, it can be used to classify sequences from unforeseen houses. This is achieved by first extracting the same meta-features in the sub-sequence and evaluating the probability using a general probability distribution. The subsequence with the final probability above a certain specific appliance threshold is accepted. The final probability is calculated using the equation given below:

$$P_{final} = P_{TOU\_DOW} \times P_{MaxPos} \times P_{MinPos} \times P_{ED} \times P_G \times P_{GR} \times P_L \quad (29)$$

P_{TOU_DOW} represents the time of use probability and day of week probability. Since distributions are used for unforeseen houses, P_{TOU_DOW} faces two problems. First, the time of use can vary between houses as we showed in our data analysis results. Second, DOW also will vary for appliances like washing machines, dishwashers. Both terms are highly related to the occupant's habits. For the first problem, it can be fairly assumed that the TOU difference will not be highly variant. For example, we showed that Dishwasher is used usually during eating times and the difference is either hour earlier or after between houses. To solve the first problem, we calculated TOU probability not only for the hour index of the event window but also for the hour earlier and hour after and took the average of three. The equation for P_{TOU} can be represented as shown below.

$$P_{TOU}(H)=(P(H-1)+P(H)+P(H+1))/3 \quad (30)$$

P_{TOU} represents TOU probability for the hour index H.

To solve the second problem, we gave less weightage to the DOW probability than others. It is implemented through P_{TOU_DOW} and can be defined as shown below.

$$P_{TOU\_DOW} = W1 \times P_{TOU} + W2 \times P_{DOW} \quad (31)$$

Where W1, W2 are weightage and its sum is equal to 1. In this way, the algorithm can include domain knowledge and at the same time robust to the occupant's habits.

P_{MaxPos} and P_{MinPos} represent the probability of local maxima/minima position in the event window actually can be local maxima/minima position for the specific appliance. since there is a possibility of two or more local maxima/minima, the algorithm calculates the probability for all local maxima/minima and takes the average out of that. We used average instead of a product to take care of the noise. Sometimes it is possible to have more local maxima/minima due to noise spikes from the other appliances. In this way, we reduced the effect of the noise.

P_{ED}, P_G, P_{GR}, PL represents energy difference probability, gradient probability, gradient ratio probability, Laplacian probability of the corresponding variables of meta-features of the event window. These terms provide the probability that these meta-features are due to a specific appliance. since each of these meta-features has more than one variable, the algorithm calculates the probability for all the variables of the meta-feature and takes the average out of it for the same reason we explained above.

The results are presented and discussed in the results section. We would like to indicate that we also tried the mean value of the event window as a meta-feature, but it reduced the performance of classification possibly due to a wide standard deviation that overlaps with other appliance distributions. We would also like to emphasize that this approach is very naïve and developed only to know the possible future exploration.

The advantages are it is simple, interpretable, requires less training time computationally less intensive than HMM and FHMM. The disadvantages are it is prone to noise and less efficient for medium-low consuming multi-state devices, the features need to be distinct enough.

4.5 Novel post-processing

In this subsection, we explained in detail our novel postprocessing system that is based on the bottom-up approach and K-NN classification.

4.5.1 Bottom-up approach

The bottom-up approach is usually employed for future prediction and it relies heavily on past data. it shows high prediction accuracy due to its utilization of a large amount of data. the underlying principle of the bottom-up approach is by modelling the subsystem in detail provides a way to obtain a complex total system by linking together all these subsystems. The top-down approach is opposite to the bottom-up approach [172]. The top-down approach starts with the complex total system and moves towards the subsystem. In the Smart grid context, the Bottom-up approach can be explained as by modelling the individual appliance's energy consumption, the total energy consumption prediction model of the house can be obtained. Various approaches have been proposed for the bottom-up approach[173]–[176]. Generally, it involves the following categories of a model to develop the bottom-up approach like weather model, socio-economic model, behavioural model, occupants and building model, temporal data like season effects, time of use, and day of the week use… Etc. we are not going to discuss in detail these algorithms as it is outside the scope of this thesis. Our work is based on the bottom-up approach proposed [177] that is based on the consumption behaviour of the residents. The idea proposed by the authors is by extracting similar days to the day of interest and evaluating appliance consumptions on those historical days it is possible to find the probability of the appliance being operated on the day of the interest. Similar day extraction is very difficult as occupant's behaviour is influenced by various factors like time, season, weekend or weekday, weather, holiday, socio-economic status…etc. the greatest degree of similarity will improve prediction accuracy. In work [177] they proposed many features to extract similar days and evaluated their prediction methodology for a single household and group of households also.

4.5.2 Failed Post Processing Methods

To improve the results of the NILM algorithm we tried two post-processing methods that failed before proposing a Similar time window approach. Both approaches were based on similar day clustering. The underlying hypothesis is that for a household if two days have a similar energy consumption pattern then for these two days, the energy is consumed by the same devices in the same manner. If we cluster similar days and extract certain information from the disaggregation results for the cluster like how many days the appliance has been operated and at what time, day of the week .etc, then with that information we can improve the classification for the particular time window. Note that the information extracted is from disaggregation results not from actual consumption data. hence the reason, the two methods we tried were only successful for the appliance for which NILM performance is high. When algorithm performance is poor, these two post-processing methods worsen the results. The steps involved in these two methods are explained below.

4.5.2.1 First method

The methodology is shown in Figure 52. The loop shown in Figure 52 will go on until it converges to the point of no change in the results. This method improved the results of the water heater since disaggregation accuracy was high for this appliance. For other appliances for which accuracy is less, the loop acts as negative feedback and worsens the results each time.

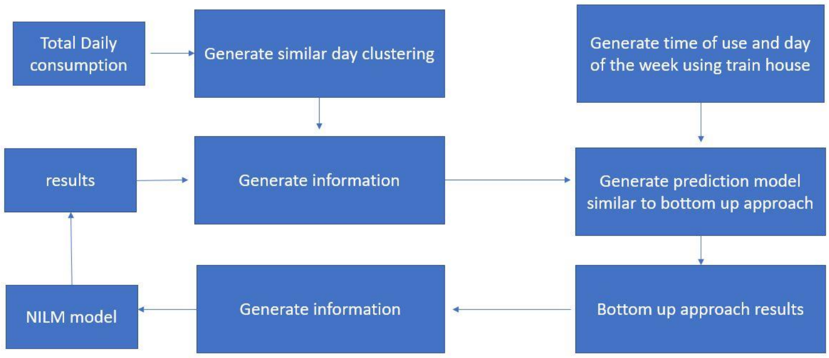

Figure 52: Methodology of the first failed post-processing method

4.5.2.2 Second method

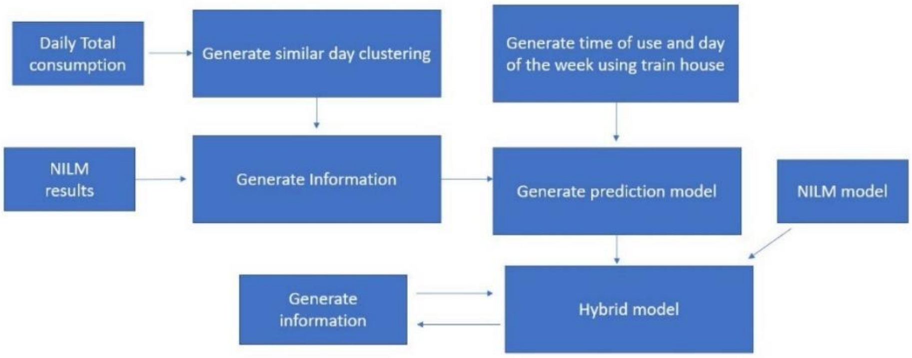

Figure 53: The methodology of the second failed post-processing method.

The methodology is shown in Figure 53. The second method is also like the first one, but we created a hybrid model for prediction since the bottom-up approach model relies on disaggregation accuracy. At the start, the hybrid model would give more weight to the NILM model but as the loop goes on its weightage reduces. This also provides the user to control the weightage depending upon the appliance as the user knows the NILM model performance for each appliance. This method also faces the same issue as the first one.

4.5.3 K-NN classification

K-NN is a supervised and instance-based learning algorithm that is first proposed in 1967 [178]. K-NN is a simple algorithm it does not create a new internal model but just stores the instances of the training data and its labels. To find the class of the new test point, K-NN categorizes the new point by a greater number of votes by its K-neighbours. For example, if the labelled dataset is represented as (X_n, Y_n) where X_n is the training data and Y_n is the class label for X_n and to find the class of new X_{test}, K-NN employs distance-based metric to find K neighbors from the training Data-set and evaluates the category votes among the neighbors. X is usually a vector. The computational complexity of the K-NN method is O(Knd) where k is the number of neighbors required, n is the number of points in space and d is the dimensionality of the point. The most popular distance-based metric for K-NN is Euclidean distance and is calculated between vectors A and B by:

$$\text{Euclidean distance (A, B)}=\sqrt[2]{(A1-B1)^2+(A2-B2)^2 \ldots\ldots\ldots\ldots..+(\mathrm{An}-\mathrm{Bn})\text{^}2} \tag{32}$$

4.5.4 Similar time window (STW) post-processing

Our hypothesis to this method is by extracting similar time windows of the test data for an instance and by evaluating its labels provided by any disaggregation algorithm it is possible to detect the wrongly labelled time window to a certain extent. Our hypothesis is based on the following fact that no single day NILM encounters are exactly the same. It is influenced by a lot of parameters. Small changes in the scenario can be tackled by NILM to its capacity. Sometimes it is possible to encounter unusual changes like if there is a new year's night party, a large number of appliances will be operating at the same usual routine operating time of the appliance A. NILM algorithms which are not capable to address this problem will be affected. It is also possible of the opposite scenario which is the usual routine of the appliance is changed to another time at which NILM wrongly classifies the appliance. there is also a possibility that if the appliance operates multiple times per day and at one time it is less affected by noise than the other time at which NILM could possibly wrongly classify the event window. Various scenarios can be described in which NILM can be wrong due to simple reasons. Indeed, it is possible to include some of these effects in the disaggregation algorithm which would increase the computational complexity and data requirements. For Low-frequency unsupervised algorithms, it is very difficult to model these effects due to fewer features available. We propose a novel post-processing method especially for low-frequency unsupervised algorithms with no special data requirements to improve the NILM.

A similar STW concept is proposed in [179] for NILM. they used a supervised method to obtain the energy consumption of the device. Their algorithm looks for the k-similar time windows in the training dataset to the test time window based on Euclidean similarity measure and assigns the device's consumption in the test window as an average of the same device's consumption in similar windows. Our algorithm is different because we do not use any ground truth data. our algorithm relies on the labels labelled by the NILM classification algorithm. To explain in simple words, if my eight out of the ten neighbours are blue then I am most likely blue than red.

The graphical representation of the steps involved in this method is shown in Figure 54. Our algorithm starts with the initial event window and looks for similar time windows in the consumption sequence. The similarity is measured using the Euclidean distance measure between the features extracted from the test window and all other windows in the sequence. Smaller distance means features are similar. Once it obtains K-STW's, it counts the labels assigned to these windows by the NILM classifier. The algorithm ignores the test window's label as it is the test point. The label of the test window is assigned based on the following conditions. First, the label count should be above a certain threshold. Second, the probability assigned to the test window by the NILM classifier should be at least 75% of the threshold probability of the appliance used by the NILM classifier. NILM classifier is trained with real appliance data. so, it is fair to assume that the event window of probability that is closer to the appliance threshold probability is possibly fallen shorter due to noise or other reasons. This is the reason for the second condition. Similarly, the algorithm iterates over all windows and post-process the labels. The algorithm is repeated until it converges. We applied this postprocessing to all three methods and compared the results. The features extracted from the time window for similarity measures are presented in the below subsection.

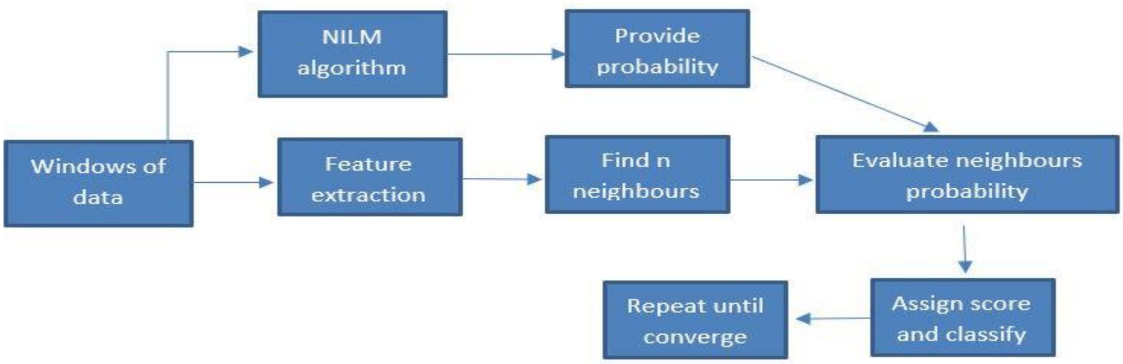

Figure 54 : Methodology of our novel post processing method.

4.5.4.1 Features extracted

To tackle the problem scenarios explained above, we selected two types of features- time-related features, consumption related feature. Time-related features help us to tackle noises whereas consumption related features help us to tackle change in user behaviour. We ensured the balance between the two to avoid dominance.

Time-related features

Time of use

It provides the first time-stamp hour value in the event window. It can be represented as below:

$$TOUn=H;\ H\epsilon\ (1,24) \tag{33}$$

Day of the week

It provides a day of the week value of the event window. Each day of the week is represented with numerical values as Monday being 0 to Sunday being 6.

Day of the week can be represented as below:

$$DOWn= D;\ D\epsilon\ (0,6) \tag{34}$$

The month of the window

This provides the month value of the event window. This ensures importance is given to the windows in the same month than other months as consumption habits change over the months more than over the days in the same month. It can be represented as below:

$$Mn=m;\ m\epsilon\ (1,12) \tag{35}$$

Consumption related features

Mean value

It provides the mean of the values in the event window. It can be fairly assumed that if two event windows have the same mean there is a possibility that these two event windows are contributed by the same appliances. It can be represented as below:

$$\mu_n \in R \tag{36}$$

Number of elements greater than a certain value

It provides the number of elements greater than a certain value in the event window. This feature prioritizes group event windows with an equal number for higher values. We chose this as this would help to differentiate the same type of appliances with different operational periods and steady-state devices with multistate devices. We can also incorporate domain knowledge in this feature. It can be represented as below:

$$N_n \in R \tag{37}$$

Dynamic time warping distance

DTW is a popular distance-based similarity measure between two-time series. Mean and the number of elements greater than certain values alone are not enough to find similarity as it does not take the value and shape of the time series into account. DTW provides those similarity values.

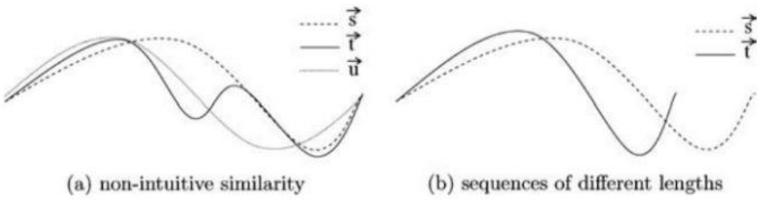

Figure 55: examples of non-intuitive and different length similar time series [180]

Simple Euclidean distance-based method provides poor similarity score between two time series sequences that may vary in time or speed. This is due to the reason that it aligns i^{th} point of one series to i^{th} point of another series. One can see it in Figure 55, how this kind of alignment could mislead us. To overcome that DTW can be used. DTW allows nonlinear alignment of points makes it possible to match similar shapes even if they are out of phase in the time axis. An example can be seen in Figure 54. Non-linear alignment without any restrictions also would provide garbage values. Certain restrictions are needed to avoid that. Details of the Various DTW algorithms are explained in [181]. we used the Python library FASTDTW to calculate DTW measure between two time-windows. The computational complexity of simple DTW for two time-series with equal length N is $O(N^2)$. FastDTW provides an optimal solution with O(N) complexity.

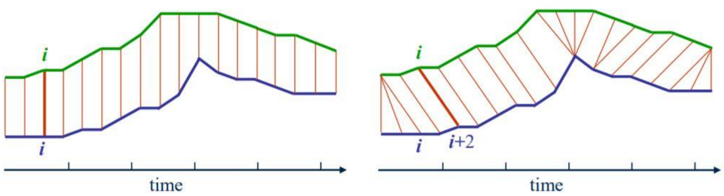

Figure 56: Difference between Euclidean distance measure and DTW measure between two time series [182]**.**

Similarity measure based on the features defined above between test point X_t and any other point X_n can be defined as below:

$$SM(X_t,X_n) = \sqrt[2]{(TOUt - TOUn)^2 + (DOWt - DOWn)^2 + (Mt - Mn)^2 + (\mu t - \mu n)^2 + (N_t - N_n)^2 + (DTWt,n)^2} \quad \ldots (38)$$

Where TOUt, TOUn= Time of Use values of test and another window

DOWt, DOWn= Day of the week values of test and another window

Mt, Mn= Month value of the test and another window

µt, µn= mean values of the test and another window

N_t, N_n= number of elements greater than a certain value in test and another window

$DTW_{t,n}$= dynamic time warping distance between test window time series and another point time series.

All the variables were normalized to equal out the influence of each variable. We would like to mention that it is a naïve approach, and the disadvantages of this method are discussed below. First, it relies on the NILM classifier accuracy as it relies on the labels provided by the classifier to post-process the results. If the classifier accuracy is lower, there is a possibility K-STW worsen the results. The computational complexity is higher as it is needed to find the distance between the test window and all other windows in the database. The second condition we used only to address the false negative problem i.e. window is assigned as not from the appliance when it is, but we did not address the opposite of this i.e. false positive problem. Another problem related to the above-mentioned point is that if the neighbours of the false positive/false negative test point are also false positive/false negative due to classifier incapacity, this algorithm cannot realize that. More sophisticated features and analysis is required to improve this method. We left that work for the future. However, our test has shown promising results that validate our concept. The results are presented and discussed in the results section.

The advantages are that it improves the accuracy of disaggregation in addition to the classifier's maximum performance. the disadvantages are it relies on classifier accuracy and computationally expensive.

Having presented the methodologies ,we present and discuss the results in next chapter.

5 Results AND Discussions

In this chapter, we present the results of the methodologies we described above. We start with HMM and move on to FHMM followed by Variational autoencoder, statistical model, and post-processing results. We analysed 50k data points for each house among which 13k-30k represents the summer consumption. We also present a time series disaggregated graph for some instances. Detailed analysis of these graphs might help us to understand and improve the model. We left this for the future as it needs more time to plan and do the systematic analysis. However, the basic analysis itself provided a few insights into our model's shortcomings. We did not present consumption values in these graphs as we developed only a basic method for value allocation and focused only on the on/off states of the appliance.

5.1 HMM Results

In this subsection, we presented and discussed the HMM results for the water heater, washing machine, clothes dryer, dishwasher.

5.1.1 Water heater

Table 13: Disaggregation results for water heater using the HMM general model

General model	House 14	House 28	House 36
Precision	0.54	0.18	0.79
Recall	0.86	0.67	0.75
F1-score	0.66	0.28	0.77

Table 14: Disaggregation results for water heater using the HMM tuned model

Tuned model	House 14	House 28	House 36
Precision	0.64	0.24	0.75
Recall	0.25	0.71	0.78
F1-score	0.36	0.36	0.76

Table 15: Disaggregation results for water heater using the HMM tuned model for the datapoints 13k-30k

Tuned model 13k-30k	House 14	House 28	House 36
Precision	0.89	0.88	0.77
Recall	0.48	0.62	0.65
F1-score	0.62	0.73	0.70

Disaggregation with the general model is presented in Table 13. With the general model, in terms of recall, the performance is good for all houses. It indicates at least 70% of water heater consumption can be detected with the general model. In terms of precision, there are no conclusive results. The performance varies between houses. This is due to other similar appliance's consumption that affects the disaggregation. Disaggregation with the tuned model is presented in Table 14. With the tuned model also we can observe similar results except for recall for house 14. From table 8, one can see that the noise percentage for the water heater in house 14 is 2% and the number of times turned-on per week is way more than the other two houses. Our exploration showed that the reason for the number of times turned on is high is that sometimes appliances were used only for short time. Due to this noise and short usage time tuned model found it difficult to disaggregate the water heater for house 14. To find the reason for low precision for some houses, we assessed the results during summer as electric space heating is the major contributor to total consumption in winter and the results are presented in Table 15. As expected, on average, the tuned model can find at least 65% of water heater consumption and at least 82% of consumption predicted is actually true.

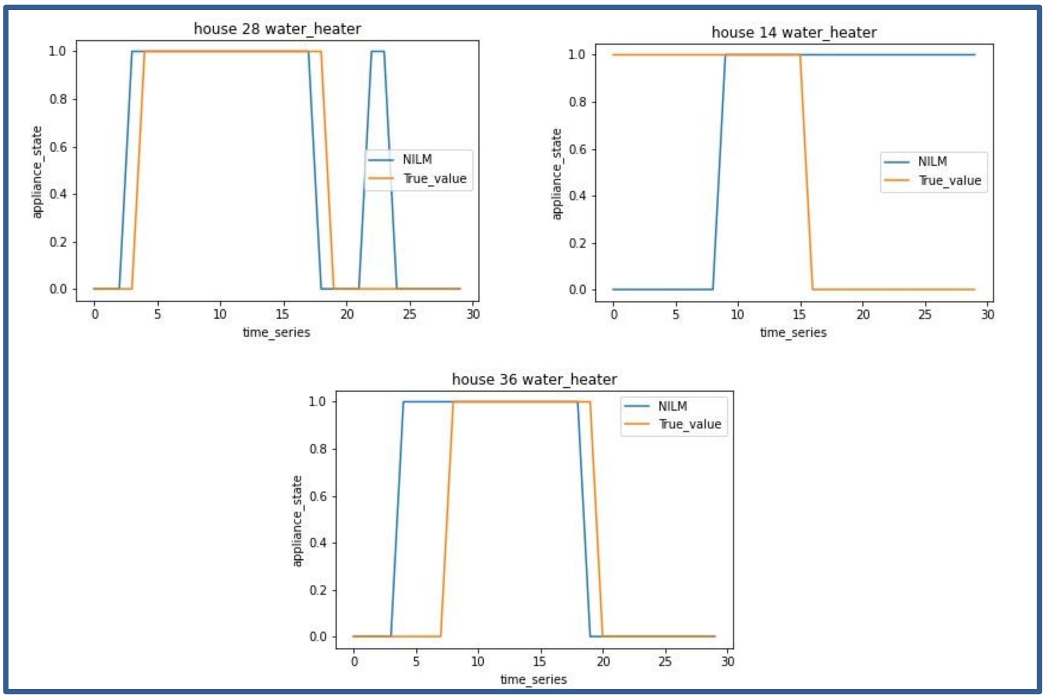

Figure 57: HMM disaggregated time series graphs for water heater

Overall, the algorithm performs very well during the summer and reasonable during winter. The tuned model improves performance. Winter disaggregation can be further improved by various methods like predicting space heating consumption and subtract it before water heater consumption, extract information from summer disaggregation and use it for winter disaggregation. Both summer and winter disaggregation can be improved by including Time of use data. The time series graphs are presented in Figure 57. The small peak in the house 28 graph is due to noise in the data window. This can be for sure discarded as noise for the water heater. The drop of nilm value before true value for house 36 is due to the reason that data interval is high and some times in that interval appliance could operate only for seconds and consumption value would be small.

5.1.2 Washing machine

Table 16: Disaggregation results for washing machine using the HMM general model

General model	House 14	House 28	House 36
Precision	0.03	0.08	0.1
Recall	0.07	0.23	0.05
F1-score	0.05	0.12	0.07

Table 17: Disaggregation results for washing machine using the HMM tuned model

Tuned model	House 14	House 28	House 36
Precision	0.02	0.04	0.22
Recall	0.08	0.34	0.18
F1-score	0.04	0.08	0.20

Table 18: Disaggregation results for washing machine using the HMM tuned model for data points 13k-30k

Tuned model 13k-30k	House 14	House 28	House 36
Precision	0.09	0.22	0.28
Recall	0.18	0.36	0.18
F1-score	0.13	0.27	0.22

The performance of disaggregation with the general model and the tuned model are presented in Tables 16 and 17. The performance is very poor for the general model. First, it is shown in the literature it is difficult to disaggregate multi-state devices. The washing machine is a multi-state device with different options. A simple general model is not enough for multiple options. Except for house 36, one can see from Table 9, the noise percentage is more than 10% when the washing machine is operating. This is the reason for low precision and recall. The tuned model increases the performance for house 36 but not for house 14 and 28. Sometimes when the general model is very poor, data collected using the general model will affect the tuning. Moreover, the data collected should be from multiple options of the washing machine which is very unlikely. Comparing summer disaggregation and overall disaggregation from Table 18 and 17, one can see performance increases.

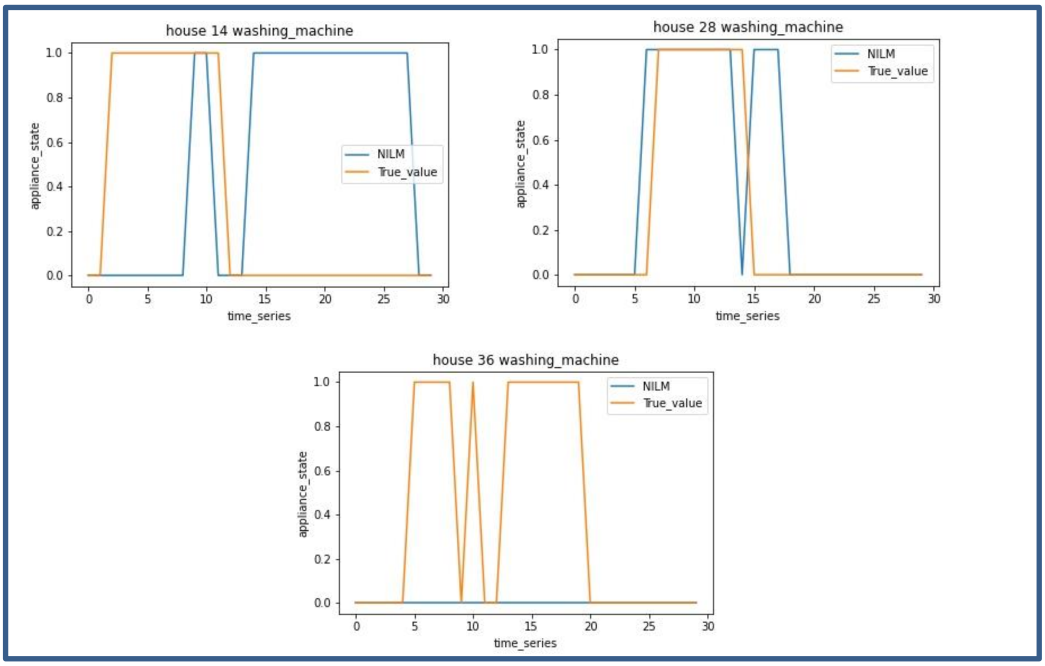

Figure 58: HMM disaggregated time series for washing machine

Overall, the model is not good enough to deploy in the real world. The model can be further improved but the complexity increases. The time series graphs are presented in Figure 58. One can see from the time-series graph of house 36 that how complex the washing machine time series could be.

5.1.3 Dishwasher

Table 19: Disaggregation results for Dishwasher using the HMM general model

General model	House 14	House 28	House 36
Precision	0.09	0.1	0.08
Recall	0.13	0.28	0.13
F1-score	0.11	0.15	0.1

Table 20:Disaggregation results for Dishwasher using HMM tuned model

Tuned model	House 14	House 28	House 36
Precision	0.10	0.08	0.08
Recall	0.12	0.27	0.12
F1-score	0.11	0.13	0.1

Table 21: Disaggregation results for Dishwasher using HMM tuned model for data points 13k-30k

Tuned model 13k-30k	House 14	House 28	House 36
Precision	0.15	0.16	0.07
Recall	0.15	0.41	0.11
F1-score	0.15	0.23	0.08

The performance of disaggregation with the general model and the tuned model are presented in Tables 19 and 20. Similar to the washing machine, Dishwasher disaggregation performance is poor. The tuned model does not improve performance. From Table 21, it can be seen Summer disaggregation is better than the general model and overall tuned model disaggregation. The reasons could be noise, poor tuning, similar device consumptions.

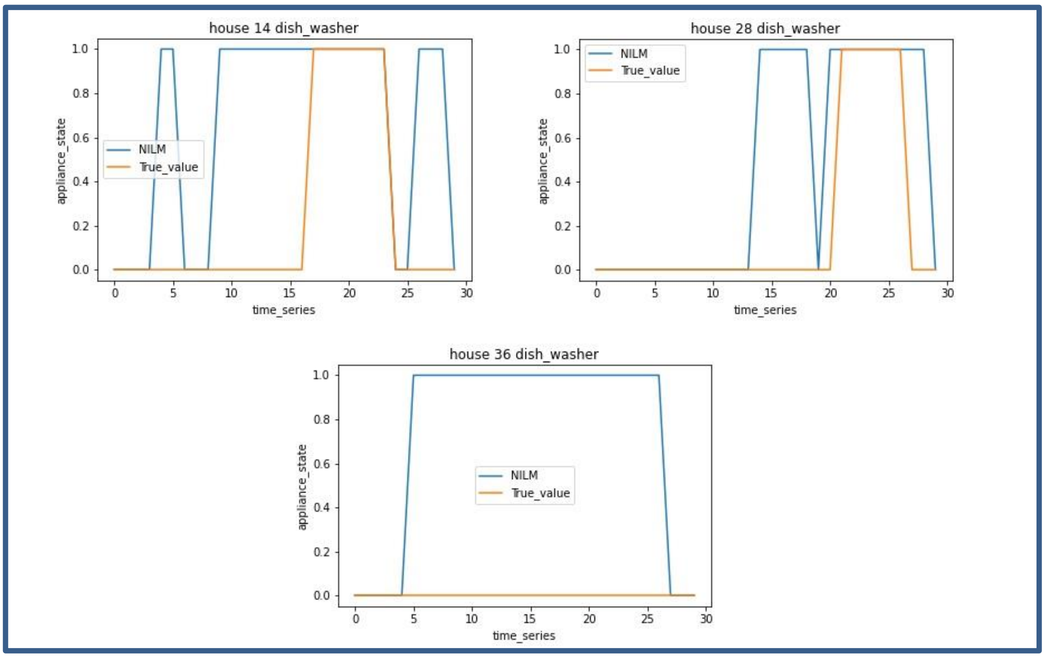

Figure 59:HMM disaggregated time series for dish washer

The time series graphs are presented in Figure 59. One can see for all graphs that NILM allots more operational period than the actual one. Data window like one for house 36 can easily be avoided by including domain knowledge which in turn would improve the precision.

5.1.4 Clothes dryer

Table 22: Disaggregation results for clothes Dryer using the HMM general model

General model	House 14	House 28	House 36
Precision	0.07	0.04	0.13
Recall	0.16	0.33	0.24
F1-score	0.10	0.08	0.15

Table 23: Disaggregation results of clothes dryer using the HMM tuned model

Tuned model	House 14	House 28	House 36
Precision	0.09	0.05	0.08
Recall	0.19	0.23	0.15
F1-score	0.12	0.08	0.10

The performance of disaggregation with the general model and the tuned model are presented in Tables 22 and 23. Both the general model and tuned model showed poor performance for the clothes dryer. The recall is better than precision. It means the models are incapable to distinguish other appliances. The recall is better for houses 28 and 36 than house 14 with the general model because the noise percentage is higher for house 14. Recall reduced with tuned model indicates that not all windows used for tuning was not from the clothes dryer. We did not assess the model for summer as the clothes dryer is used frequently during winter. Adding domain knowledge like time of use, duration of use might improve precision.

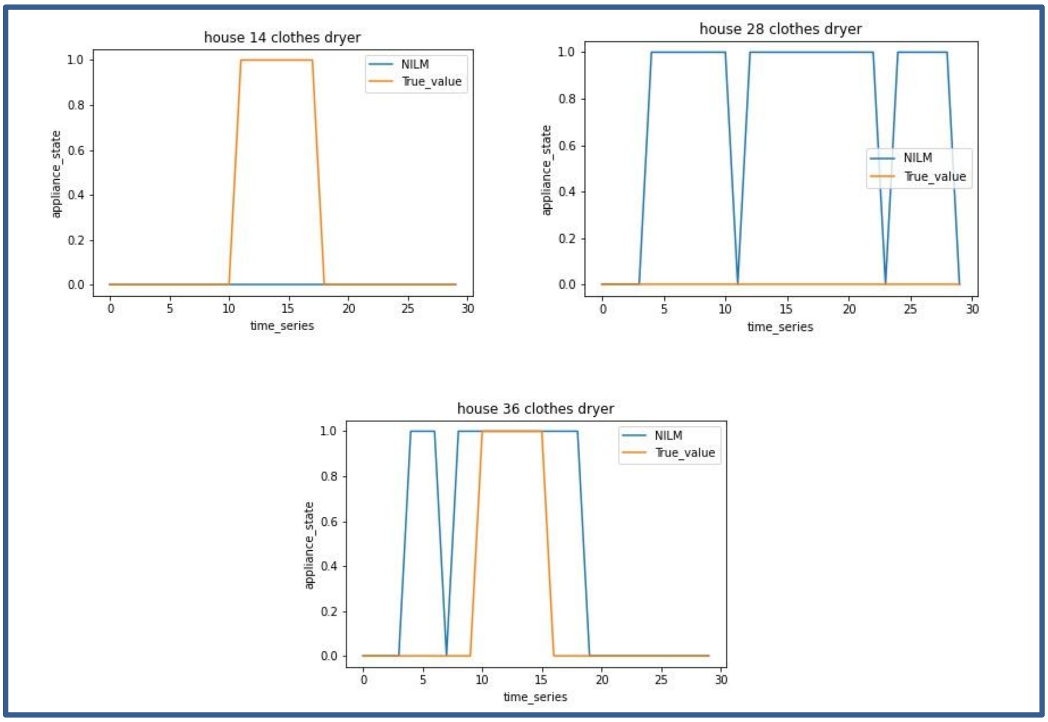

Figure 60: HMM disaggregated time series for clothes dryer

The time-series graphs for the clothes dryer are presented in Figure 60 . one could tell that the time series for house 28 is out of the ordinary. The reason being that it is possible only if the washing machine and clothes dryer both has the same operational period in order to immediately transfer it and dry it or if cloths dryer capacity is very small. This is where little information about the clothes dryer obtained from the owner could be useful to avoid false positives like this.

5.1.5 Electric space heater

Disaggregation results of ESH using both general and tuned models are presented in Tables 24 and 25. Both general and tuned models performed well except for house 36. This validates that the General prior model can be applied to FHMM for a set of appliances. The poor results for house 36 are due to the reason that at any point the maximum energy consumption is less than 300Wh which is way less than the ESH consumption in the training data. Probably that house has a single space heater. Poor precision can be explained by the fact that other appliances' energy consumptions are also in the same range and ESH has been used way less in house 36 than the other two houses. However, the performance of the model for house 14 and house 28 is promising.

Table 24: Disaggregation results of Electric Space Heaters using FHMM General model

General model	House 14	House 28	House 36
Precision	0.69	0.79	0.07
Recall	0.79	0.91	0.29
F1-score	0.73	0.84	0.12

Table 25:Disaggregation results of Electric Space Heaters using FHMM tuned model

Tuned model	House 14	House 28	House 36
Precision	0.72	0.8	0.11
Recall	0.80	0.72	0.35
F1-score	0.75	0.76	0.16

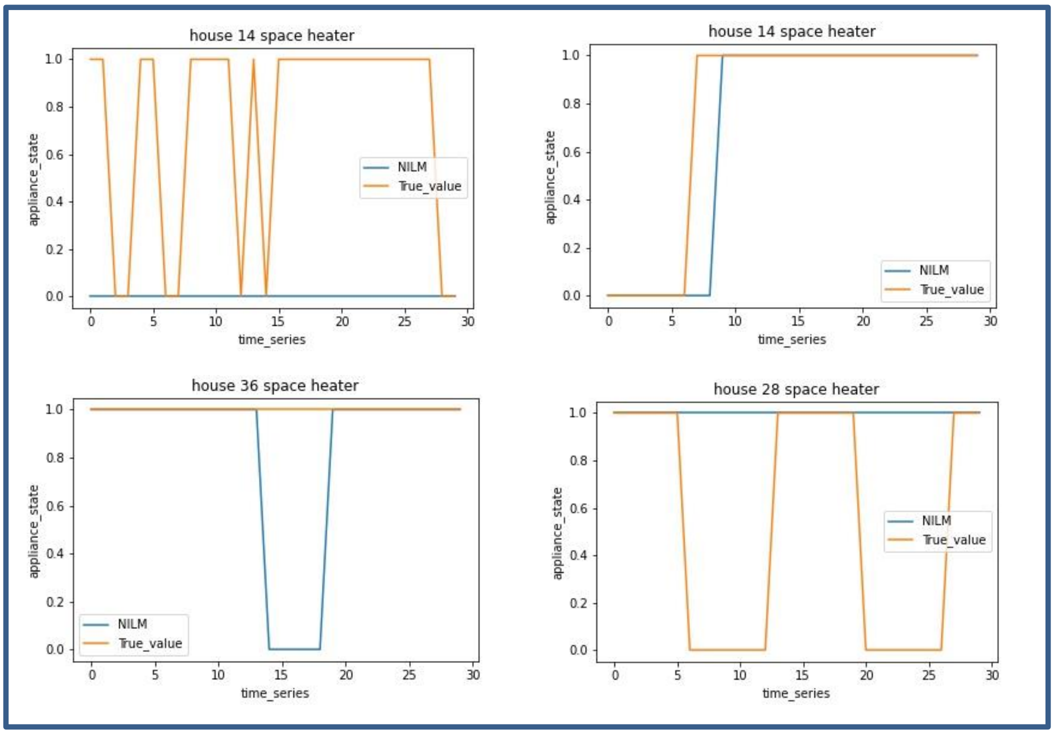

Figure 61: FHMM disaggregated time series for space heaters.

The time series graphs are presented in Figure 61. House 14 graph shows the time series complexity due to the 10-minute data interval. The drop in the NILM value for house 36 is due to the reason that we subtract the minimum value as constant load as some times the minimum value itself from the appliance of interest.

5.2 Variational Auto-encoder results

5.2.1 Water heater

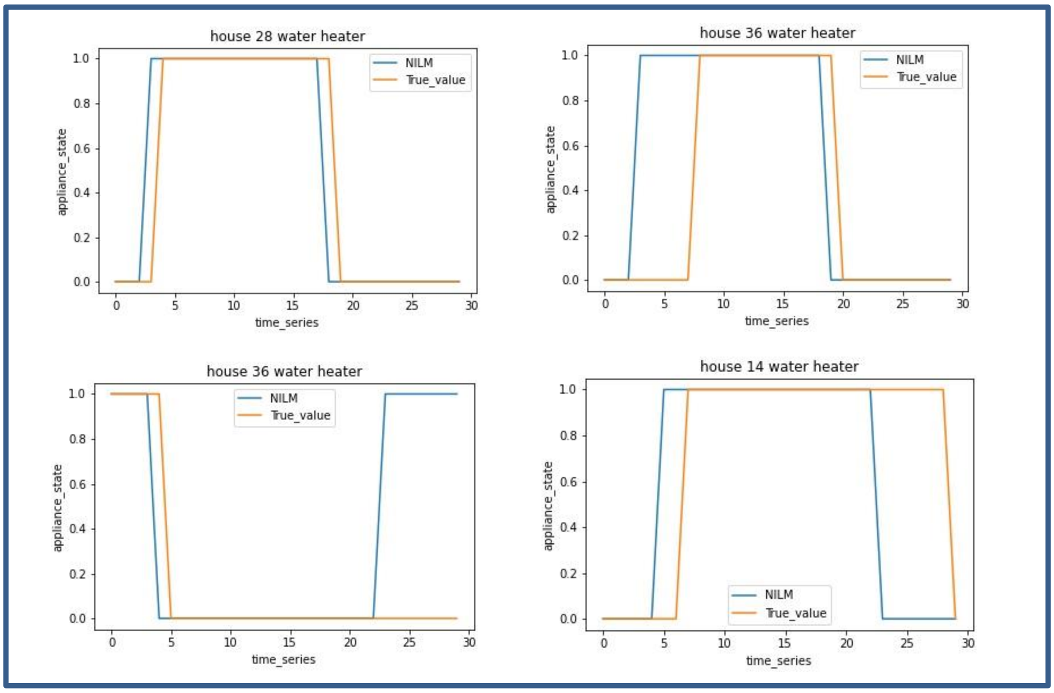

Figure 62: VAE disaggregated time series graphs for water heater

Disaggregation results of water heater consumption using VAE are presented in Table 26. Comparing tables 26 and 14, one can find the F1 score trend is similar between houses. Overall VAE performs better than the tuned HMM model. It means it is better than HMM in sensing noise and other appliances' consumptions. Table 27 shows that the time of use integration increases the performance of the disaggregation except for house 14. The reason is that occupant's habits changes between houses. The summer disaggregation presented in Table 28 shows the model's extraordinary capabilities. It shows that the model can find at least on average 73% of water heater consumption and on average 89% of the predicted value is actually true. We would like to indicate that TOU is not used in HMM that is the reason we compared tables 14 and 26 and not others. The time series graphs are presented in Figure 62. The false value in side to true value in the graphs is promising as it can be avoided easily by developing the sophisticated value allocation module.

Table 26: Disaggregation results of the water heater using the VAE model

VAE	House 14	House 28	House 36
Precision	0.84	0.51	0.82
Recall	0.66	0.74	0.86
F1-score	0.74	0.61	0.84

Table 27: Disaggregation results of the water heater using the VAE + TOU model

VAE+ TOU	House 14	House 28	House 36
Precision	0.87	0.55	0.89
Recall	0.60	0.74	0.85
F1-score	0.71	0.63	0.87

Table 28: Disaggregation results of the water heater using the VAE + TOU model for data points 13k-30k

VAE+ TOU 13k-30k	House 14	House 28	House 36
Precision	0.95	0.87	0.84
Recall	0.61	0.77	0.84
F1-score	0.74	0.82	0.84

5.2.2 Washing machine

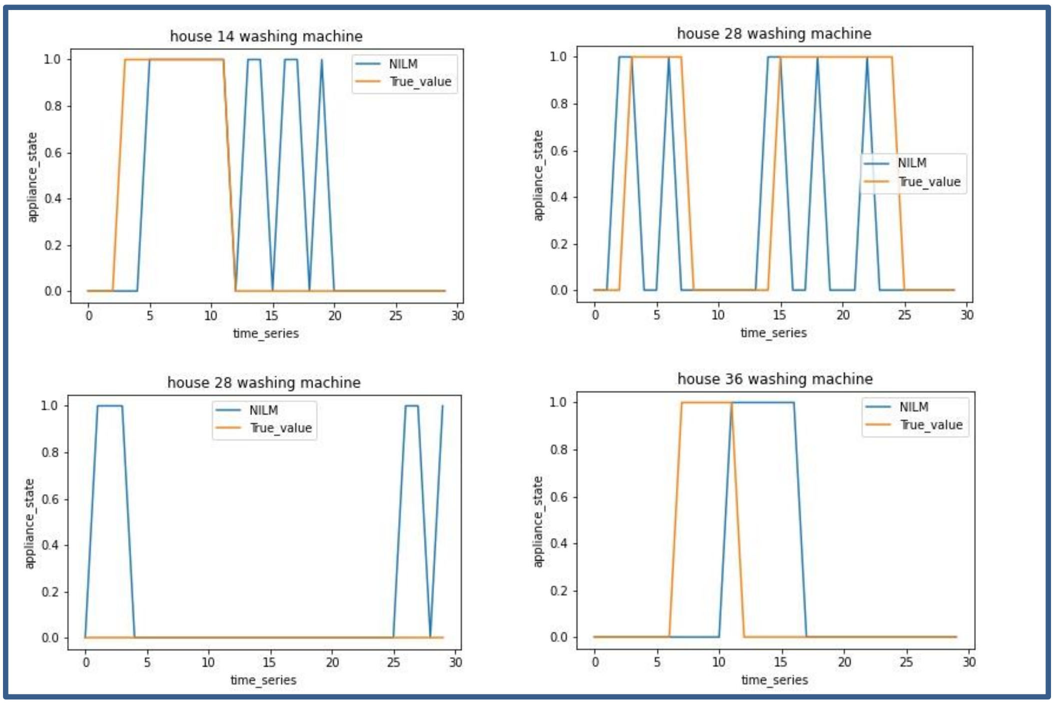

Figure 63: VAE disaggregated time series graphs for washing machine.

Disaggregation results of washing machine consumption using VAE are presented in Table 29. The washing machine model showed poor performance. VAE model performs better than the HMM model except for house 28. This indicates both models have different capabilities to tackle different NILM issues. The time of use model results presented in Table 30 shows that it did not improve the performance much. This is due to the reason that the washing machine is not used daily and usage depends on the occupant's convenience, need. Table 31 shows that summer disaggregation is much better than overall disaggregation. House 14 and house 36 results show that the VAE model is capable to model multiple options of the washing machine, can handle noise, and differentiate other appliances better than the tuned HMM model. The time series graphs are presented in Figure 63.

Table 29: Disaggregation results of the washing machine using the VAE model

VAE	House 14	House 28	House 36
Precision	0.11	0.08	0.35
Recall	0.38	0.20	0.27
F1-score	0.17	0.12	0.31

Table 30:Disaggregation results of the washing machine using the VAE + TOU model

VAE+ TOU	House 14	House 28	House 36
Precision	0.11	0.10	0.36
Recall	0.31	0.19	0.28
F1-score	0.16	0.13	0.32

Table 31:Disaggregation results of the washing machine using the VAE + TOU model for data points 13k-30k

VAE+ TOU 13k-30k	House 14	House 28	House 36
Precision	0.20	0.1	0.46
Recall	0.35	0.09	0.30
F1-score	0.26	0.1	0.37

5.2.3 Clothes dryer

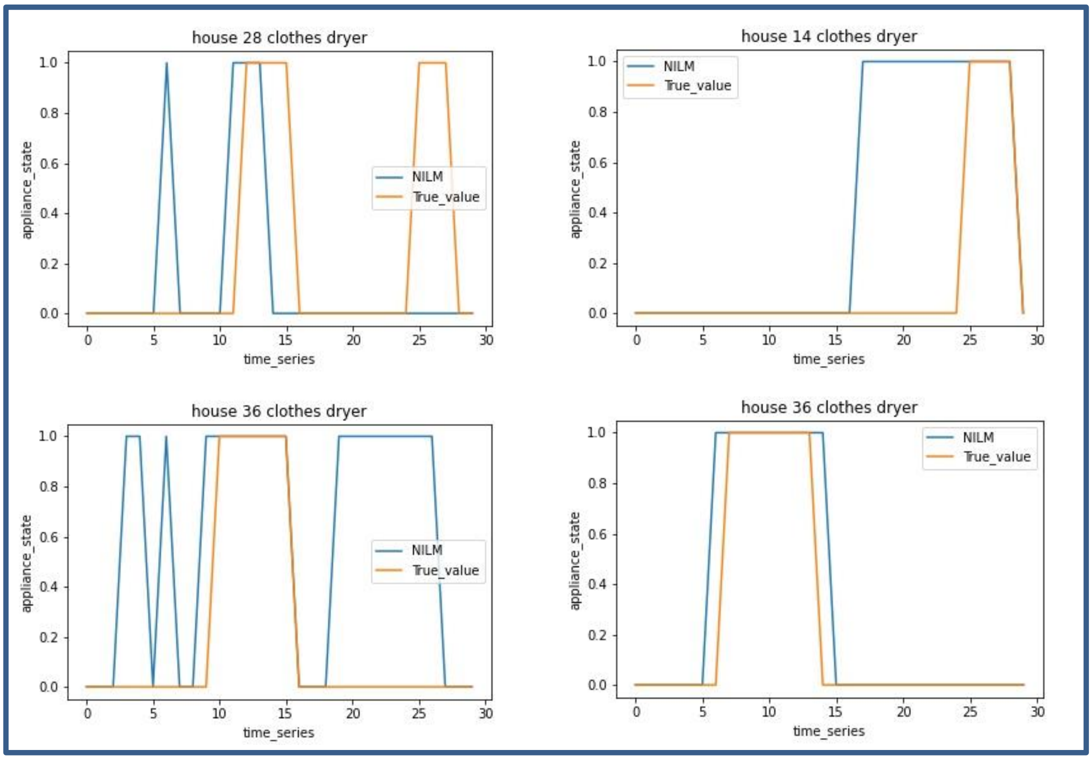

Figure 64: VAE disaggregated time series graphs for clothes dryer

Table 32: Disaggregation results of Clothes dryer using VAE model

VAE	House 14	House 28	House 36
Precision	0.09	0.11	0.15
Recall	0.31	0.23	0.52
F1-score	0.14	0.15	0.24

Disaggregation results of clothes dryer consumption using VAE are presented in Table 32. VAE clothes dryer results are better than HMM clothes dryer results and the trend is similar between houses. It can be seen in Table 33 that Time of use increases the performance of disaggregation. Comparing recall results of the HMM model and VAE model for house14 indicates the model is better at handling noise. House 28 recall is lower than house 14 recall which indicates, the model suffers from something else than noise, possibly operational period which is much lower than the train houses' clothes drier's operational period. The time series graphs are presented in Figure 64.

Table 33: Disaggregation results of Clothes dryer using VAE + TOU model

VAE+ TOU	House 14	House 28	House 36
Precision	0.11	0.15	0.23
Recall	0.31	0.22	0.48
F1-score	0.16	0.18	0.31

5.2.4 Dishwasher

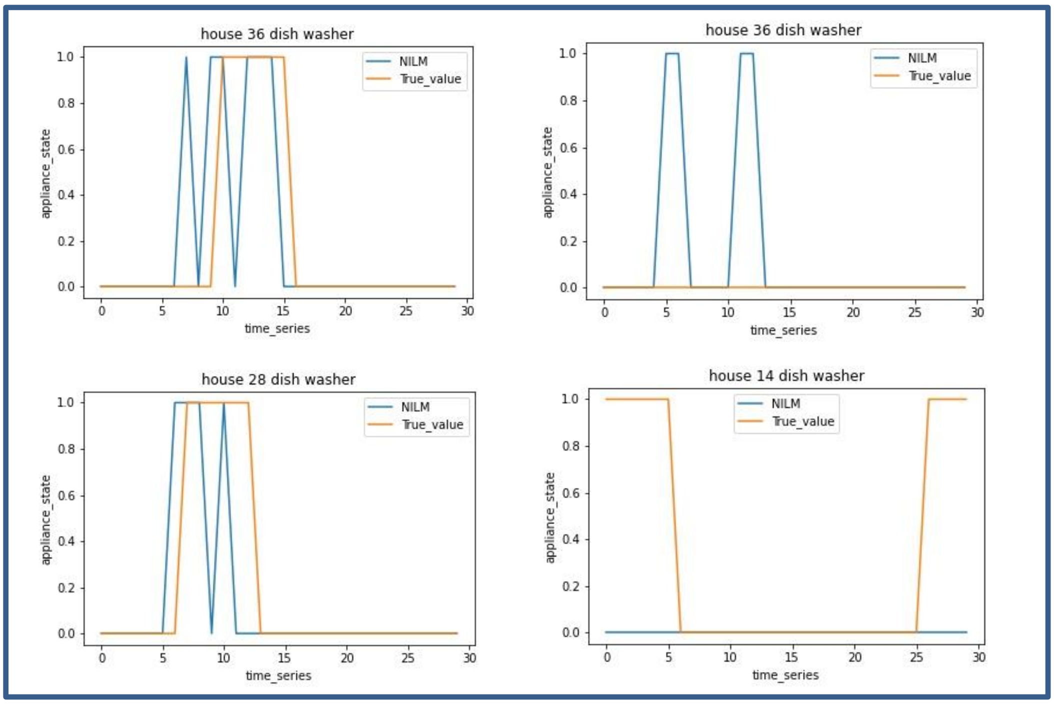

Figure 65: VAE disaggregated time series graphs for dish washer

Table 34: Disaggregation results of Dishwasher using VAE model

VAE	House 14	House 28	House 36
Precision	0.04	0.29	0.28
Recall	0.08	0.43	0.28
F1-score	0.06	0.35	0.28

Table 35: Disaggregation results of Dishwasher using VAE + TOU model

VAE+ TOU	House 14	House 28	House 36
Precision	0.05	0.40	0.34
Recall	0.07	0.40	0.18
F1-score	0.06	0.40	0.24

Overall Disaggregation results of dishwasher consumption using VAE are presented in Table 34. Again, the VAE model performed better than the HMM model for the dishwasher. Dishwasher results were even better than washing machine results for the VAE model except for house 14. The noise could be the reason for house 14 poor results because the noise percentage is 12 for house 14. Table 35 shows that there are no conclusive results that indicate the time of use improves performance. It sure does reduces false positives. The time series graphs are presented in Figure 65. Summer disaggregation results are presented in Table 36.

Table 36: Disaggregation results of Dishwasher using VAE + TOU model for data points 13k-30k

VAE+ TOU 13k-30k	House 14	House 28	House 36
Precision	0.13	0.66	0.19
Recall	0.10	0.40	0.09
F1-score	0.11	0.5	0.12

5.2.5 Electric Space Heater

Table 37: Disaggregation results of Electric space heater using VAE model

VAE	House 14	House 28	House 36
Precision	0.78	0.89	0.08
Recall	0.53	0.44	0.19
F1-score	0.63	0.59	0.12

Table 38: Disaggregation results of Electric space heater using VAE + TOU model

VAE+ TOU	House 14	House 28	House 36
Precision	0.80	0.91	0.09
Recall	0.50	0.41	0.19
F1-score	0.62	0.57	0.12

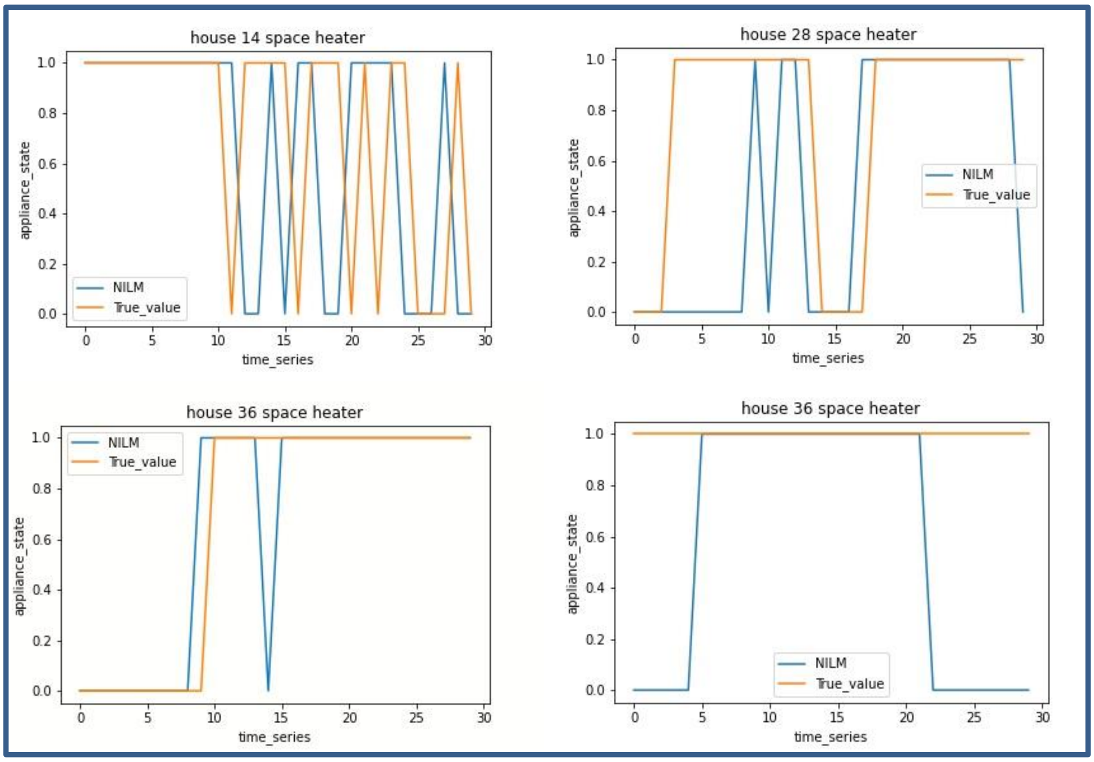

Figure 66: VAE disaggregated time series graphs for space heater

Disaggregation results of ESH using VAE and VAE+TOU model are presented in Tables 30 and 31. The performance of each model is not bad and poorer than the FHMM model. The precision is actually better than the FHMM model which is promising because if 80% of disaggregation is actually true, we can trust the model for demand-side management. Our post-processing method has proven to improve the recall score. Combined with that it is possible to create a better disaggregation algorithm for ESH. Though the FHMM model has better performance it is computationally very intensive than VAE. Further exploration into test data and results might provide insight into a low recall score. The time series graphs are presented in Figure 66. One can see in the house 14 time-series graph that the model was able to capture the complex dynamics.

5.3 Statistical rules-based Temporal classifier results

5.3.1 Water heater

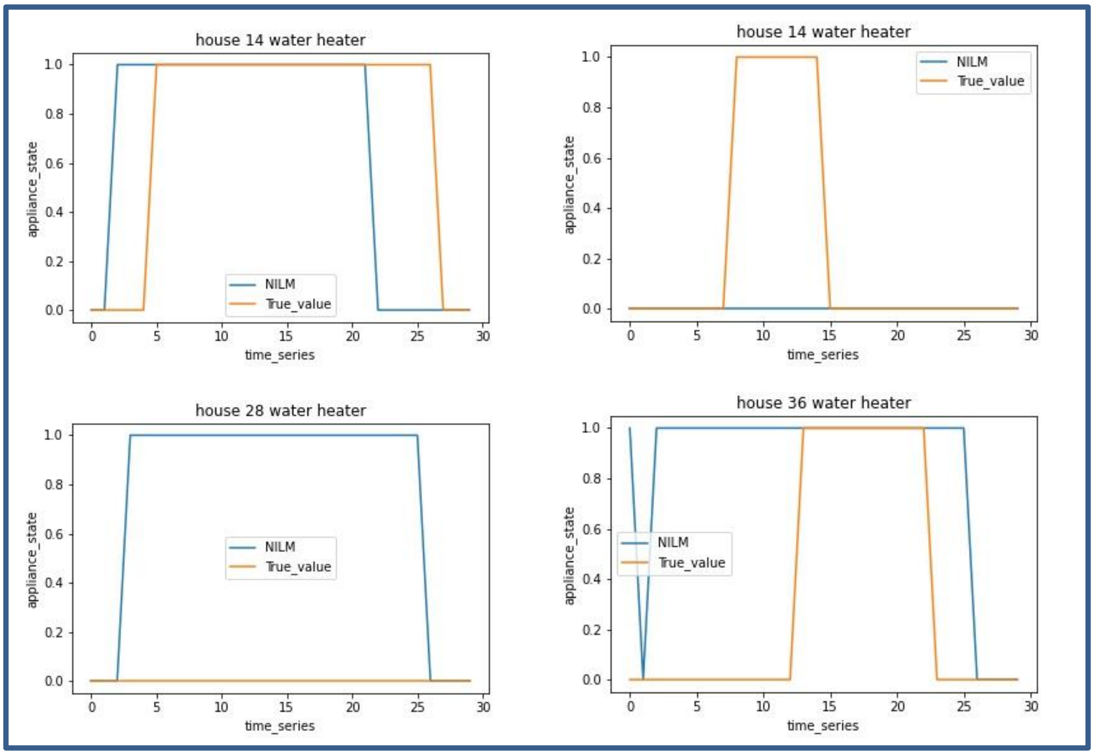

Figure 67: statistical method disaggregated time series graphs for water heater

The overall and summer disaggregation results are presented in Tables 39 and 40. The statistical rules-based model's performance is similar to HMM and poorer than the VAE model. Unlike two other models, the summer disaggregation does not improve the results which indicates that the model is suffering from a different issue than the other two models. Note that disaggregation was carried out with general meta-features, it is possible to tune the general meta-features to a specific house and it might improve the results. The time series graphs are presented in Figure 67.

Table 39: Disaggregation results of the water heater using statistical rules-based model

Statistical rules	House 14	House 28	House 36
Precision	0.81	0.16	0.62
Recall	0.37	0.27	0.97
F1-score	0.51	0.20	0.76

Table 40: Disaggregation results of the water heater using the statistical rules-based model for data points 13k-30k

Statistical rules(13k-30k)	House 14	House 28	House 36
Precision	0.87	0.68	0.58
Recall	0.26	0.15	0.92
F1-score	0.40	0.25	0.71

5.3.2 Washing machine

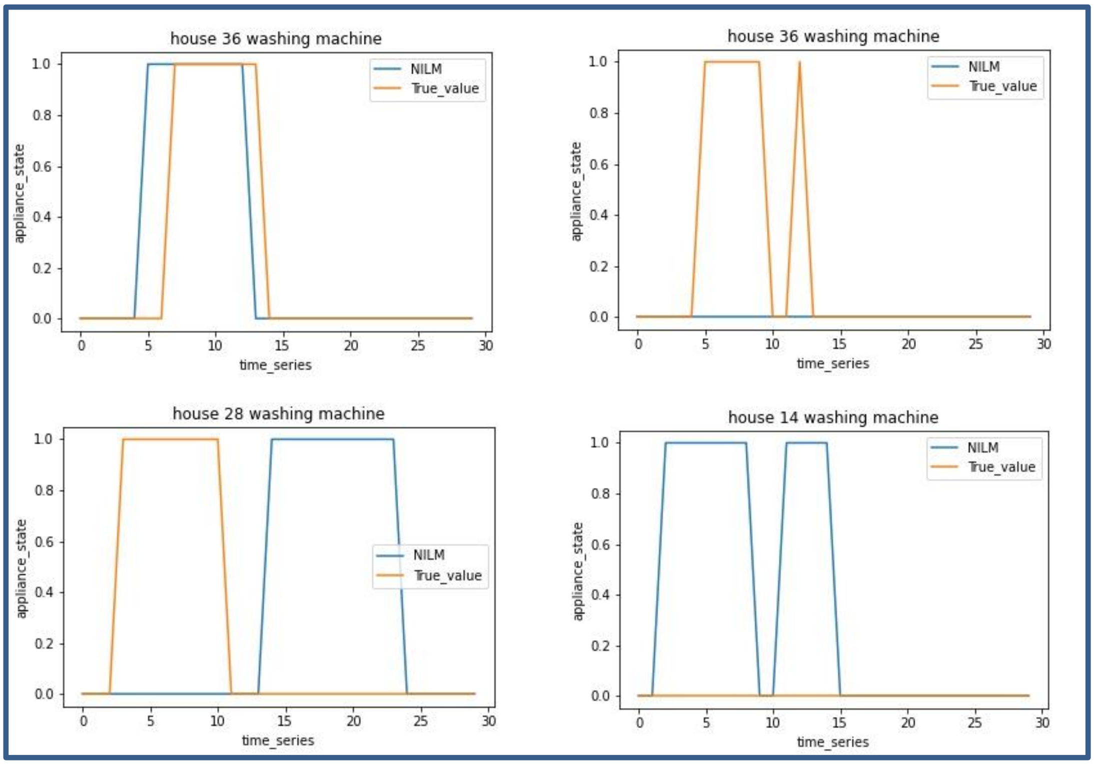

Figure 68:Statistical method disaggregated time series graphs for washing machine

Table 41: Disaggregation results of the washing machine using statistical rules-based model

Statistical rules	House 14	House 28	House 36
Precision	0.11	0.02	0.27
Recall	0.44	0.29	0.49
F1-score	0.18	0.05	0.35

Table 42: Disaggregation results of the washing machine using the statistical rules-based model for data points 13k-30k

Statistical rules(13k-30k)	House 14	House 28	House 36
Precision	0.18	0.06	0.30
Recall	0.47	0.11	0.50
F1-score	0.26	0.08	0.37

The overall and summer disaggregation results are presented in Tables 41 and 42. The washing machine model performs better than the HMM model and similar to the VAE model. The recall is actually better for three houses than the VAE model recall. It is promising because when we develop a method to subtract the other appliances' consumption there is a possibility that precision might improve. It also indicates the features we proposed is not enough to distinguish other similar devices. The time series graphs are presented in Figure 68.

5.3.3 Dishwasher

Table 43: Disaggregation results of Dishwasher using statistical rules-based model

Statistical rules	House 14	House 28	House 36
Precision	0.1	0.08	0.06
Recall	0.19	0.22	0.15
F1-score	0.14	0.11	0.09

Table 44: Disaggregation results of Dishwasher using the statistical rules-based model for data points 13k-30k

Statistical rules(13k-30k)	House 14	House 28	House 36
Precision	0.11	0.21	0.08
Recall	0.09	0.19	0.13
F1-score	0.09	0.2	0.1

The overall and summer disaggregation results are presented in Tables 43 and 44. Overall, the model's performance is poor for the dishwasher. The performance is similar to the HMM model and poorer than the VAE model. Like water heater, summer disaggregation does not improve the results except for house 28. The poor Precision score indicates features were not distinguishing enough. The time series graphs are presented in Figure 69.

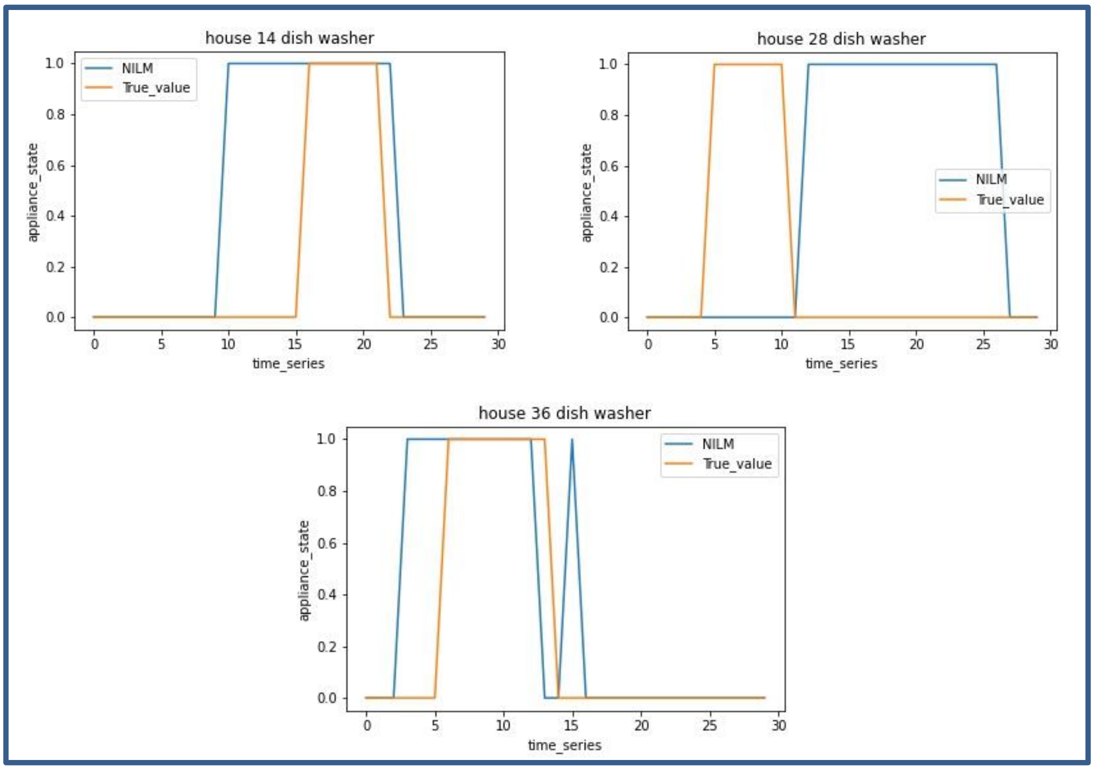

Figure 69: Statistical method disaggregated time series graphs for dish washer

5.3.4 Clothes dryer

Table 45: Disaggregation results of clothes dryer using statistical rules-based model

Statistical rules	House 14	House 28	House 36
Precision	0.07	0.07	0.09
Recall	0.37	0.46	0.49
F1-score	0.11	0.10	0.15

The overall disaggregation results are presented in Table 45. The model's performance is very poor for the clothes dryer. However, this model has better recall than the HMM and VAE model's recall and provides a similar F1 score. The time series graphs are presented in Figure 70.

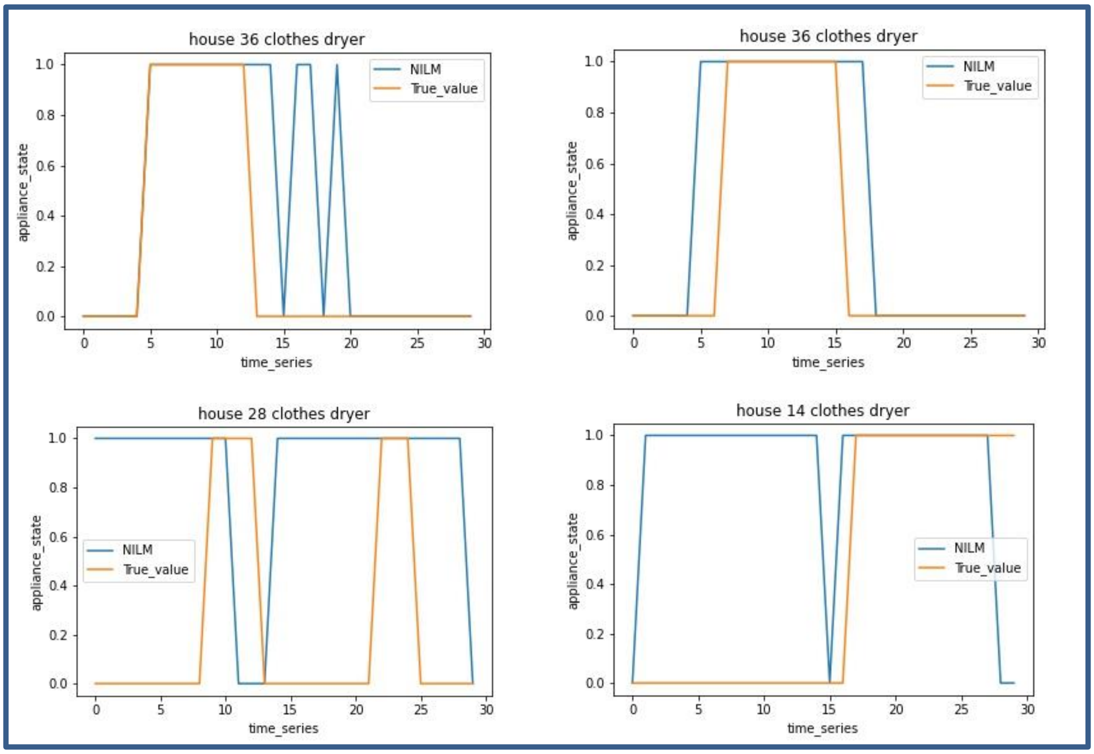

Figure 70: Statistical method disaggregated time series graphs for clothes dryer

5.3.5 Electric Space heater

Table 46: Disaggregation results of Electric Space Heater using statistical rules-based model

Statistical rules	House 14	House 28	House 36
Precision	0.62	0.75	0.05
Recall	0.89	0.73	0.52
F1-score	0.73	0.74	0.1

The disaggregation results of the electric space heater using the statistical rules-based method are presented in Table 46. the model shows equivalent performance to the FHMM model and better performance than the VAE model. This model is computationally less intensive than the FHMM model. The results of the water heater and ESH have proven promising for statistical rules-based methods and further improvement might provide a better model suitable for real-world deployment. The time series graphs are presented in Figure 71.

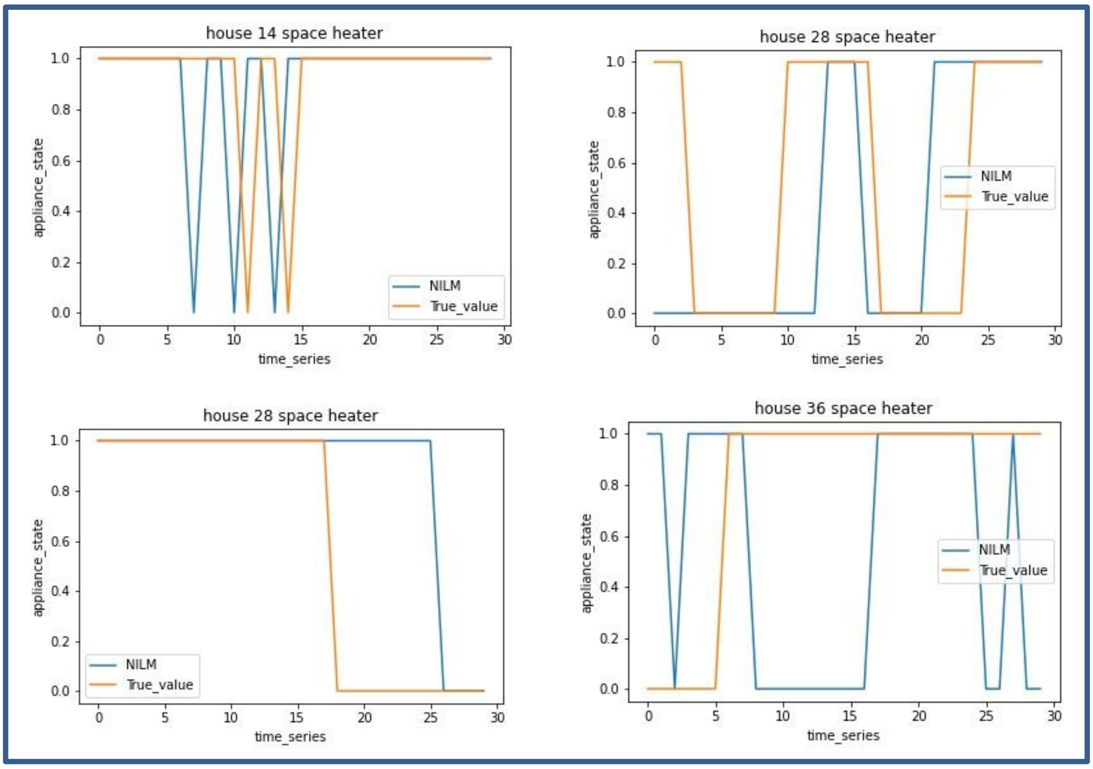

Figure 71: Statistical method disaggregated time series graphs for space heater

5.4 Postprocessing(pp)

5.4.1 Water heater

Table 47: Postprocessing results of the water heater using Similar time window method

VAE+ TOU 13k-30k	House14 before PP	House14 after PP	House28 before pp	House28 after PP	House36 before pp	House36 after PP
Precision	0.95	0.87	0.87	0.98	0.84	0.86
Recall	0.61	0.81	0.77	0.98	0.84	0.95
F1-score	0.74	0.83	0.82	0.98	0.84	0.9

we applied our postprocessing method to VAE results and are presented in Table 47. In Table 47 one can see the overall improvement in the F1 score for all three houses. Particularly for house 28, the improvements are remarkable. F1 score increases from 0.81 to 0.98. one can see there is a decrease in precision score for house 14. We explained this situation before in the previous chapter that sometimes if the NILM algorithm accepts similar false windows our postprocessing method can not know that and it will accept more similar false windows.

5.4.2 Washing machine

Table 48: Postprocessing results of the washing machine using Similar time window method

VAE+ TOU 13k-30k	House14 before PP	House14 after PP	House28 before pp	House28 after PP	House36 before pp	House36 after PP
Precision	0.20	0.2	0.1	0.12	0.46	0.4
Recall	0.35	0.67	0.09	0.27	0.30	0.49
F1-score	0.26	0.31	0.1	0.17	0.37	0.43

To show our method applies to all appliances, we applied our method to washing machine VAE results and are presented in Table 48. One can see improvements in the F1 score for all houses even when their F1 score is below 0.5. In particular, house 28 results show the strength of our proposed method. Even when precision and recall were below 0.1 our method was able to improve the results.

5.4.3 House 14 water heater

Table 49:Postprocessing results of different NILM model results

House14 13k-30k	VAE before PP	VAE after PP	HMM before pp	HMM after pp	Statistical rules before pp	Statistical rules after pp
Precision	0.95	0.87	0.89	0.82	0.87	0.8
Recall	0.61	0.81	0.48	0.71	0.26	0.71
F1-score	0.74	0.83	0.63	0.76	0.40	0.75

To show that our method is common to all NILM algorithms, we applied the postprocessing to House 14 water heater results of all three methods explored in this thesis are presented in Table 49. There is an improvement in the F1 score for all three methods.

Overall, it can be said that our postprocessing method can improve the results. It can also be interpreted from all three tables that postprocessing results depend upon the appliance of interest and other appliances' characteristics in the house and also NILM algorithms capability. One can also see there is not much improvement in precision and sometimes precision score decreases after postprocessing. We explained the reason in the previous chapter, and we will focus on this problem in future works.

Having presented and discussed the results for selected appliances with the methodologies explored in this work,we conclude the report by presenting the conclusion and future works in next section.

6 Conclusion and Future works

We explored three different models for NILM and tested for different appliances. In summary, the results showed that models can be deployed in the real-world for demand-side management only for the water heater and electric space heating. The methods can not be completely non-intrusive and high performing. The results show that without any information about appliances the performances are poor. So it is reasonable to consider methods that rely on some kind of intrusions like a survey or Human input methods like interactive learning. All three methods performed poorly for multi-state devices like washing machines, dishwashers, clothes dryers. One of the methods we proposed VAE for NILM showed similar performance for the water heater and electric heating and higher performance for multi-state devices than the state of the art HMM method for NILM. Another method we proposed also shows similar performance for the water heater and electric heating to the other two methods. When it comes to multi-state devices the model did not perform better than VAE. However, some promising results for multistate devices and good results for the water heater and electric heating shows that model can be improved further. Our greatest contribution to NILM with this thesis is our novel post-processing method. We validated our hypothesis and proved that results can be improved. Our post-processing method's ability to improve Recall more provides flexibility to the NILM algorithm to tighten the threshold thereby it can reduce the false positives. It also allows us to reduce the training time or development time as usually fine-tuning of algorithms usually gives little benefits.

- Since we treated each appliance model-independent and applied total energy as input the same window has been accepted by multiple appliance models. We discussed two methods already proposed in [9][106] to overcome this situation. We plan to explore these two options in the future. Besides, we are also proposing a new method to overcome this situation. The models will be treated independently but with a constraint, the single window can belong only to one model. This will be enforced by comparing the probability output of the window by appliance models and make the window belong to the appliance with maximum probability.
- We also noticed that HMM is prone to noise from other appliances. In [11] they proposed a method to overcome this noise problem using a forward likelihood estimation algorithm. We plan to examine this method and we believe this method can be further improved by incorporating domain knowledge. For example, we can find an almost clear signal using an identification algorithm that would provide a lot of details about the appliance which can be used to create the filter algorithm.
- We also noticed the tuning worsen the general model for multi-state devices. This is due to false windows used for tuning. We plan to improve the identification results by incorporating domain knowledge like Time of use, Day of use or maybe use VAE for identification as it had better recall than the HMM model.
- Disaggregation using HMM can be improved in so many ways proposed in the literature. We plan to start by incorporating the general time of use model, general appliance correlation, occupant's behavior. Using initial estimate and postprocessing we plan to tune these features also for a specific house.
- Hierarchical HMM can be used to model multiple options multi-state devices like a washing machine. It allows us to model each option of the appliance.
- All the above-mentioned plans can also be tested for FHMM. We tested FHMM only for space heaters. We plan to test it for a different set of appliances and validate its ability to act as a virtual sub-meter and explore the possibility to use it as a feature generator for further disaggregation.
- Computational complexity is very high for Gibbs sampling we will explore other inference methods.
- Instead of two models for VAE based disaggregation, we can create one model for both reconstruction error and classification. It can be done by adding one more output node in addition to nodes for reconstruction. And it is also possible to have a single model for all appliances instead of a single model for each appliance.
- We will also explore the possibility of reducing the training time and include domain knowledge into the VAE model.

- Since VAE is known for its new content generation, we will explore the ways to use this for NILM benefits. Possibly to get more training data or with some occupant’s survey, it can create the possible signal dictionary for the appliance in the house.
- Our statistical rules-based method is very naïve. We plan to explore more features to improve the results. We also plan to incorporate domain knowledge and study feature importance.
- Our postprocessing can be further improved by including more features and domain knowledge like appliance correlations, appliance usage patterns. For example, it might not be possible to have a washing machine being used more than 15 times in a week. This kind of knowledge can be used to further improve the results. We can apply the data mining techniques to create such information and develop some rules for each appliance using data-driven methods.
- We also plan to explore the ensemble of the three methods to take advantage of the strengths of each method.
- Finally, we plan to create a complete system from data processing to postprocessing with these methods and simulate the demand side management and examine the efficiency and viability for real-world deployment.

We conclude this report by pointing out that we proposed two new methods for unsupervised NILM of which one has shown superior performance than state of art NILM algorithm and another one has shown some promising results to explore further. Our novel postprocessing has shown tremendous ability and has opened up so many ways to improve the NILM system in the future.

REFERENCES

[1] D. Hafemeister, "Sustainable Energy—Without the Hot Air Sustainable Energy—Without the Hot AirDavid J.C.MacKay. 372 pp. UIT Cambridge Ltd., Cambridge, UK, 2009. Price: $33.00(paper) ISBN 978-0-9544529-3-3. ," *Am. J. Phys.*, 2010.

[2] IEA, "World Energy Outlook 2019," 2019. [Online]. Available: https://www.iea.org/reports/world-energy-outlook-2019/electricity. [Accessed: 15-Mar-2020].

[3] T. Logenthiran, D. Srinivasan, and T. Z. Shun, "Demand side management in smart grid using heuristic optimization," *IEEE Trans. Smart Grid*, 2012.

[4] J. Revuelta Herrero *et al.*, "Non intrusive load monitoring (NILM): A state of the art," in *Advances in Intelligent Systems and Computing*, 2017.

[5] K. Carrie Armel, A. Gupta, G. Shrimali, and A. Albert, "Is disaggregation the holy grail of energy efficiency? The case of electricity," *Energy Policy*, 2013.

[6] K. Basu, A. Hably, V. Debusschere, S. Bacha, G. J. Driven, and A. Ovalle, "A comparative study of low sampling non intrusive load dis-aggregation," in *IECON Proceedings (Industrial Electronics Conference)*, 2016.

[7] European Commision, "Benchmarking Smart Metering deployment in the EU 27," 2014. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52014SC0189&from=EN. [Accessed: 15-Mar-2020].

[8] IEA, "The impacts of Covid-19 crisis on global energy demand and CO2 emissions," 2020. [Online]. Available: https://www.iea.org/reports/global-energy-review-2020/electricity#abstract. [Accessed: 15-Mar-2020].

[9] O. Parson, "UNSUPERVISED TRAINING METHODS FOR NON-INTRUSIVE APPLIANCE LOAD MONITORING FROM SMART METER DATA," University of Southampton, 2014.

[10] C. Shin, S. Rho, H. Lee, and W. Rhee, "Data requirements for applying machine learning to energy disaggregation," *Energies*, 2019.

[11] O. Parson, S. Ghosh, M. Weal, and A. Rogers, "Non-intrusive load monitoring using prior models of general appliance types," in *Proceedings of the National Conference on Artificial Intelligence*, 2012.

[12] K. Basu, V. Debusschere, and S. Bacha, "Residential appliance identification and future usage prediction from smart meter," in *IECON Proceedings (Industrial Electronics Conference)*, 2013.

[13] M. C. Baechler and H. Hao, "Business Case for Nonintrusive Load Monitoring," *Pacific Northwest Natl. Lab.*, 2016.

[14] G. W. Hart, "Nonintrusive Appliance Load Monitoring," *Proc. IEEE*, 1992.

[15] R. McWilliam and A. Purvis, "Potential uses of embedded RFID in appliance identification.," *Int. Appl. Manuf.*, 2006.

[16] C. Klemenjak and P. Goldsborough, "Non-intrusive load monitoring: A review and outlook," in *Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI)*, 2016.

[17] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science (80-.).*, 1983.

[18] S. N. Patel, T. Robertson, J. A. Kientz, M. S. Reynolds, and G. D. Abowd, "At the Flick of a Switch: Detecting and Classifying Unique Electrical Events on the Residential Power Line (Nominated for the Best Paper Award)," in *UbiComp 2007: Ubiquitous Computing*, 2007.

[19] K. H. Tiedemann, “Using conditional demand analysis to estimate residential energy use and energy savings,” *Proc. CDEEE*, 2007.

[20] S. Makonin, “Approaches to Non-Intrusive Load Monitpring (NILM) in the Home,” SIMON FRASER UNIVERSITY, 2012.

[21] A. Ruano, A. Hernandez, J. Ureña, M. Ruano, and J. Garcia, “NILM techniques for intelligent home energy management and ambient assisted living: A review,” *Energies*. 2019.

[22] A. U. Haq and H. A. Jacobsen, “Prospects of appliance-level load monitoring in off-The-shelf energy monitors: A technical review,” *Energies*, 2018.

[23] S. Platforms, E. Non-intrusive, and L. M. Systems, “Hardware and software platforms to deploy and evaluate non-intrusive load monitoring systems,” Universidade da MADEIRA, 2016.

[24] S. Drenker and A. Kader, “Nonintrusive monitoring of electric loads,” *IEEE Comput. Appl. Power*, 1999.

[25] A. I. Cole and A. Albicki, “Data extraction for effective non-intrusive identification of residential power loads,” in *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, 1998.

[26] A. I. Cole and A. Albicki, “Algorithm for non-intrusive identification of residential appliances,” in *Proceedings - IEEE International Symposium on Circuits and Systems*, 1998.

[27] F. Sultanem, “Using appliance signatures for monitoring residential loads at meter panel level,” *IEEE Trans. Power Deliv.*, 1991.

[28] L. K. Norford and S. B. Leeb, “Non-intrusive electrical load monitoring in commercial buildings based on steady-state and transient load-detection algorithms,” *Energy Build.*, 1996.

[29] D. Srinivasan, W. S. Ng, and A. C. Liew, “Neural-network-based signature recognition for harmonic source identification,” *IEEE Trans. Power Deliv.*, 2006.

[30] C. Laughman *et al.*, “Power signature analysis,” *IEEE Power Energy Mag.*, 2003.

[31] M. Berges, E. Goldman, H. S. Matthews, and L. Soibelman, “Learning systems for electric consumption of buildings,” in *Proceedings of the 2009 ASCE International Workshop on Computing in Civil Engineering*, 2009.

[32] M. E. Berges, E. Goldman, H. S. Matthews, and L. Soibelman, “Enhancing electricity audits in residential buildings with nonintrusive load monitoring,” *J. Ind. Ecol.*, 2010.

[33] M. Dong, P. C. M. Meira, W. Xu, and C. Y. Chung, “Non-intrusive signature extraction for major residential loads,” *IEEE Trans. Smart Grid*, 2013.

[34] J. Powers, B. Margossian, and B. Smith, “Using a Rule-Based Algorithm to Disaggregate End-Use Load Profiles from Premise-Level Data,” *IEEE Comput. Appl. Power*, 1991.

[35] M. B. Figueiredo, A. De Almeida, and B. Ribeiro, “An experimental study on electrical signature identification of non-intrusive load monitoring (NILM) systems,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011.

[36] M. L. Marceau and R. Zmeureanu, “Nonintrusive load disaggregation computer program to estimate the energy consumption of major end uses in residential buildings,” *Energy Convers. Manag.*, 2000.

[37] M. Baranski and J. Voss, “Non-intrusive appliance load monitoring based on an optical sensor,” in *2003 IEEE Bologna PowerTech - Conference Proceedings*, 2003.

[38] M. Baranski and J. Voss, “Detecting patterns of appliances from total load data using a dynamic programming approach,” in *Proceedings - Fourth IEEE International Conference on Data Mining, ICDM*

2004, 2004.

[39] M. Baranski and J. Voss, “Genetic algorithm for pattern detection in NIALM systems,” in *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 2004.

[40] A. G. Ruzzelli, C. Nicolas, A. Schoofs, and G. M. P. O’Hare, “Real-time recognition and profiling of appliances through a single electricity sensor,” in *SECON 2010 - 2010 7th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, 2010.

[41] S. B. Leeb, S. R. Shaw, and J. L. Kirtley, “Transient Event Detection in Spectral Envelope Estimates for Nonintrusive Load Monitoring,” *IEEE Trans. Power Deliv.*, 1995.

[42] S. R. Shaw, S. B. Leeb, L. K. Norford, and R. W. Cox, “Nonintrusive load monitoring and diagnostics in power systems,” *IEEE Trans. Instrum. Meas.*, 2008.

[43] K. D. Lee, S. B. Leeb, L. K. Norford, P. R. Armstrong, J. Holloway, and S. R. Shaw, “Estimation of variable-speed-drive power consumption from harmonic content,” *IEEE Trans. Energy Convers.*, 2005.

[44] W. L. Chan, A. T. P. So, and L. L. Lai, “Harmonics load signature recognition by wavelets transforms,” in *DRPT2000. International Conference on Electric Utility Deregulation and Restructuring and Power Technologies.*, 2002.

[45] 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 *FIIW 2012 - 2012 Future of Instrumentation International Workshop Proceedings*, 2012.

[46] M. Gray and W. G. Morsi, “Application of wavelet-based classification in non-intrusive load monitoring,” in *Canadian Conference on Electrical and Computer Engineering*, 2015.

[47] J. M. Gillis, S. M. Alshareef, and W. G. Morsi, “Nonintrusive load monitoring using wavelet design and machine learning,” *IEEE Trans. Smart Grid*, 2016.

[48] J. M. Gillis and W. G. Morsi, “Non-Intrusive Load Monitoring Using Semi-Supervised Machine Learning and Wavelet Design,” *IEEE Trans. Smart Grid*, 2017.

[49] S. M. Tabatabaei, S. Dick, and W. Xu, “Toward Non-Intrusive Load Monitoring via Multi-Label Classification,” *IEEE Trans. Smart Grid*, 2017.

[50] H. H. Chang, “Non-intrusive demand monitoring and load identification for energy management systems based on transient feature analyses,” *Energies*, 2012.

[51] H. H. Chang, K. L. Chen, Y. P. Tsai, and W. J. Lee, “A new measurement method for power signatures of nonintrusive demand monitoring and load identification,” in *IEEE Transactions on Industry Applications*, 2012.

[52] Y. C. Su, K. L. Lian, and H. H. Chang, “Feature selection of non-intrusive load monitoring system using STFT and wavelet transform,” in *Proceedings - 2011 8th IEEE International Conference on e-Business Engineering, ICEBE 2011*, 2011.

[53] H. Y. Lam, G. S. K. Fung, and W. K. Lee, “A novel method to construct taxonomy electrical appliances based on load signatures,” *IEEE Trans. Consum. Electron.*, 2007.

[54] T. Hassan, F. Javed, and N. Arshad, “An empirical investigation of V-I trajectory based load signatures for non-intrusive load monitoring,” *IEEE Trans. Smart Grid*, 2014.

[55] L. Du, D. He, R. G. Harley, and T. G. Habetler, “Electric load classification by binary voltage-current trajectory mapping,” *IEEE Trans. Smart Grid*, 2016.

[56] S. Inagaki, T. Egami, T. Suzuki, H. Nakamura, and K. Ito, “Nonintrusive appliance load monitoring based

on integer programming," *Electr. Eng. Japan (English Transl. Denki Gakkai Ronbunshi)*, 2011.

[57] T. D. Huang, W. S. Wang, and K. L. Lian, "A New Power Signature for Nonintrusive Appliance Load Monitoring," *IEEE Trans. Smart Grid*, 2015.

[58] D. TESHOME, T. D. Huang, and K.-L. Lian, "A Distinctive Load Feature Extraction Based on Fryze's Time-domain Power Theory," *IEEE Power Energy Technol. Syst. J.*, 2016.

[59] Jian Liang, S. Ng, G. Kendall, and J. Cheng, "Load signature study <inf>¡</inf>V part I: Basic concept, structure and methodology," *IEEE Trans. Power Deliv.*, vol. 25, no. 2, 2010.

[60] J. Gao, E. C. Kara, S. Giri, and M. Berges, "A feasibility study of automated plug-load identification from high-frequency measurements," in *2015 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2015*, 2016.

[61] S. Lin, L. Zhao, F. Li, Q. Liu, D. Li, and Y. Fu, "A nonintrusive load identification method for residential applications based on quadratic programming," *Electr. Power Syst. Res.*, 2016.

[62] K. D. Anderson, M. E. Berges, A. Ocneanu, D. Benitez, and J. M. F. Moura, "Event detection for Non Intrusive load monitoring," in *IECON Proceedings (Industrial Electronics Conference)*, 2012.

[63] C. C. Yang, C. S. Soh, and V. V. Yap, "Comparative study of event detection methods for nonintrusive appliance load monitoring," in *Energy Procedia*, 2014.

[64] K. N. Trung *et al.*, "Event Detection and Disaggregation Algorithms for NIALM System," in *NILM workshop*, 2014.

[65] B. Wild, K. S. Barsim, and B. Yang, "A new unsupervised event detector for non-intrusive load monitoring," in *2015 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2015*, 2016.

[66] A. Marchiori, D. Hakkarinen, Q. Han, and L. Earle, "Circuit-level load monitoring for household energy management," *IEEE Pervasive Comput.*, 2011.

[67] L. Farinaccio and R. Zmeureanu, "Using a pattern recognition approach to disaggregate the total electricity consumption in a house into the major end-uses," *Energy Build.*, 1999.

[68] H. Najmeddine *et al.*, "State of art on load monitoring methods," in *PECon 2008 - 2008 IEEE 2nd International Power and Energy Conference*, 2008.

[69] T. Kato, H. S. Cho, D. Lee, T. Toyomura, and T. Yamazaki, "Appliance recognition from electric current signals for information-energy integrated network in home environments," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2009.

[70] A. Cole and A. Albicki, "Nonintrusive identification of electrical loads in a three-phase environment based on harmonic content," in *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, 2000.

[71] J. Li, S. West, and G. Platt, "Power decomposition based on SVM regression," in *Proceedings of 2012 International Conference on Modelling, Identification and Control, ICMIC 2012*, 2012.

[72] W. K. Lee, G. S. K. Fung, H. Y. Lam, F. H. Y. Chan, and M. Lucente, "Exploration on Load Signatures," *Electr. Eng.*, 2004.

[73] S. Gupta, M. S. Reynolds, and S. N. Patel, "ElectriSense: Single-point sensing using EMI for electrical event detection and classification in the home," in *UbiComp'10 - Proceedings of the 2010 ACM Conference on Ubiquitous Computing*, 2010.

[74] N. Sadeghianpourhamami, J. Ruyssinck, D. Deschrijver, T. Dhaene, and C. Develder, "Comprehensive

feature selection for appliance classification in NILM," *Energy Build.*, 2017.

[75] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar, "Non-intrusive Load Monitoring approaches for disaggregated energy sensing: A survey," *Sensors (Switzerland)*. 2012.

[76] M. Zeifman and K. Roth, "Nonintrusive appliance load monitoring: Review and outlook," in *Digest of Technical Papers - IEEE International Conference on Consumer Electronics*, 2011.

[77] H. H. Chang, H. T. Yang, and C. L. Lin, "Load identification in neural networks for a non-intrusive monitoring of industrial electrical loads," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008.

[78] M. Hazas, A. Friday, and J. Scott, "Look back before leaping forward: Four decades of domestic energy inquiry," *IEEE Pervasive Comput.*, 2011.

[79] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han, "Unsupervised disaggregation of low frequency power measurements," in *Proceedings of the 11th SIAM International Conference on Data Mining, SDM 2011*, 2011.

[80] M. Zeifman, "Disaggregation of home energy display data using probabilistic approach," *IEEE Trans. Consum. Electron.*, 2012.

[81] M. Wytock and J. Z. Kolter, "Contextually supervised source separation with application to energy disaggregation," in *Proceedings of the National Conference on Artificial Intelligence*, 2014.

[82] M. Zhuang, M. Shahidehpour, and Z. Li, "An Overview of Non-Intrusive Load Monitoring: Approaches, Business Applications, and Challenges," in *2018 International Conference on Power System Technology, POWERCON 2018 - Proceedings*, 2019.

[83] W. Kong, Z. Y. Dong, D. J. Hill, F. Luo, and Y. Xu, "Improving Nonintrusive Load Monitoring Efficiency via a Hybrid Programing Method," *IEEE Trans. Ind. Informatics*, 2016.

[84] M. Z. A. Bhotto, S. Makonin, and I. V. Bajić, "Load Disaggregation Based on Aided Linear Integer Programming," *IEEE Trans. Circuits Syst. II Express Briefs*, 2017.

[85] D. Egarter, A. Sobe, and W. Elmenreich, "Evolving non-intrusive load monitoring," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013.

[86] H. Wang, W. Yang, T. Chen, and Q. Yang, "An optimal load disaggregation method based on power consumption pattern for low sampling data," *Sustain.*, 2019.

[87] S. Giri, M. Bergés, and A. Rowe, "Towards automated appliance recognition using an EMF sensor in NILM platforms," *Adv. Eng. Informatics*, 2013.

[88] S. Barker, M. Musthag, D. Irwin, and P. Shenoy, "Non-intrusive load identification for smart outlets," in *2014 IEEE International Conference on Smart Grid Communications, SmartGridComm 2014*, 2015.

[89] M. Figueiredo, A. de Almeida, and B. Ribeiro, "Home electrical signal disaggregation for non-intrusive load monitoring (NILM) systems," *Neurocomputing*, 2012.

[90] J. Z. Kolter, S. Batra, and A. Y. Ng, "Energy disaggregation via discriminative sparse coding," in *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010, NIPS 2010*, 2010.

[91] T. Zia, D. Bruckner, and A. Zaidi, "A hidden Markov model based procedure for identifying household electric loads," in *IECON Proceedings (Industrial Electronics Conference)*, 2011.

[92] F. J. Ferrández-Pastor, H. Mora-Mora, J. L. Sánchez-Romero, M. Nieto-Hidalgo, and J. M. García-Chamizo,

"Interpreting human activity from electrical consumption data using reconfigurable hardware and hidden Markov models," *J. Ambient Intell. Humaniz. Comput.*, 2017.

[93] D. Egarter, V. P. Bhuvana, and W. Elmenreich, "PALDi: Online load disaggregation via particle filtering," *IEEE Trans. Instrum. Meas.*, 2015.

[94] Q. Wu and F. Wang, "Concatenate convolutional neural networks for non-intrusive load monitoring across complex background," *Energies*, 2019.

[95] M. S. Tsai and Y. H. Lin, "Modern development of an Adaptive Non-Intrusive Appliance Load Monitoring system in electricity energy conservation," *Appl. Energy*, 2012.

[96] P. Paulo, "Applications of Deep Learning Techniques on Nilm.Ph.D. Thesis, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil," pp. 4–6, 2016.

[97] J. D. Kelly, "Disaggregation of Domestic Smart Meter Energy Data," University of London, 2016.

[98] H. He, Z. Liu, R. Jiao, and G. Yan, "A novel nonintrusive load monitoring approach based on linear-chain conditional random fields," *Energies*, 2019.

[99] Jian Liang, S. Ng, G. Kendall, and J. Cheng, "Load signature study <inf>¡</inf>V part II: Disaggregation framework, simulation and applications," *IEEE Trans. Power Deliv.*, vol. 25, no. 2, 2010.

[100] S. Alshareef and W. G. Morsi, "Application of wavelet-based ensemble tree classifier for non-intrusive load monitoring," in *2015 IEEE Electrical Power and Energy Conference: Smarter Resilient Power Systems, EPEC 2015*, 2016.

[101] H. Gonçalves, A. Ocneanu, M. Bergés, and R. H. Fan, "Unsupervised disaggregation of appliances using aggregated consumption data," *Environ. Eng.*, 2011.

[102] P. Winkler, G. Le Ray, and P. Pinson, "Unsupervised Energy Disaggregation: From Sparse Signal Approximation to Community Detection," *IEEE Trans. Smart Grid*, no. iii, pp. 1–8, 2019.

[103] J. Z. Kolter and T. Jaakkola, "Approximate inference in additive factorial HMMs with application to energy disaggregation," in *Journal of Machine Learning Research*, 2012.

[104] M. J. Johnson and A. S. Willsky, "Bayesian nonparametric Hidden semi-Markov models," *J. Mach. Learn. Res.*, 2013.

[105] S. Pattem, "Unsupervised disaggregation for non-intrusive load monitoring," in *Proceedings - 2012 11th International Conference on Machine Learning and Applications, ICMLA 2012*, 2012.

[106] M. Figueiredo and A. De Almeida, "On the Optimization of Appliance Loads Inferred by Probabilistic Models," *NILM Work. 2014*, no. 1, 2014.

[107] J. Liao, G. Elafoudi, L. Stankovic, and V. Stankovic, "Power Disaggregation for Low-sampling Rate Data," *NILM Work. 2014*, 2014.

[108] K. S. Barsim, R. Streubel, and B. Yang, "An Approach for Unsupervised Non-Intrusive Load Monitoring of Residential Appliances," in *Proceedings of the 48th International Universities' Power Engineering Conference (UPEC)*, 2013.

[109] L. Liu, J. Ding, J. Zhong, X. Fu, and Y. Lv, "An unsupervised model for classification and recognition of household appliances," *J. Comput. Inf. Syst.*, 2014.

[110] M. Figueiredo, B. Ribeiro, and A. De Almeida, "Electrical signal source separation via nonnegative tensor factorization using on site measurements in a smart home," *IEEE Trans. Instrum. Meas.*, 2014.

[111] H. Shao, M. Marwah, and N. Ramakrishnan, "A temporal motif mining approach to unsupervised energy

disaggregation: Applications to residential and commercial buildings," in *Proceedings of the 27th AAAI Conference on Artificial Intelligence, AAAI 2013*, 2013.

[112] L. Wang, X. Luo, and W. Zhang, "Unsupervised energy disaggregation with factorial hidden Markov models based on generalized backfitting algorithm," in *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, 2013.

[113] G. Elafoudi, L. Stankovic, and V. Stankovic, "Power disaggregation of domestic smart meter readings using dynamic time warping," in *ISCCSP 2014 - 2014 6th International Symposium on Communications, Control and Signal Processing, Proceedings*, 2014.

[114] K. Srinivasarengan, Y. G. Goutam, M. G. Chandra, and S. Kadhe, "A framework for non intrusive load monitoring using bayesian inference," in *Proceedings - 7th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, IMIS 2013*, 2013.

[115] R. Bonfigli, S. Squartini, M. Fagiani, and F. Piazza, "Unsupervised algorithms for non-intrusive load monitoring: An up-to-date overview," in *2015 IEEE 15th International Conference on Environment and Electrical Engineering, EEEIC 2015 - Conference Proceedings*, 2015.

[116] V. Stankovic, J. Liao, and L. Stankovic, "A graph-based signal processing approach for low-rate energy disaggregation," in *IEEE SSCI 2014 - 2014 IEEE Symposium Series on Computational Intelligence - CIES 2014: 2014 IEEE Symposium on Computational Intelligence for Engineering Solutions, Proceedings*, 2014.

[117] H. H. Chang, K. L. Lian, Y. C. Su, and W. J. Lee, "Power-spectrum-based wavelet transform for nonintrusive demand monitoring and load identification," *IEEE Trans. Ind. Appl.*, 2014.

[118] G. Y. Lin, S. C. Lee, J. Y. J. Hsu, and W. R. Jih, "Applying power meters for appliance recognition on the electric panel," in *Proceedings of the 2010 5th IEEE Conference on Industrial Electronics and Applications, ICIEA 2010*, 2010.

[119] W. L. Chan, A. T. P. So, and L. L. Lai, "Wavelet feature vectors for neural network based harmonics load recognition," in *IEE Conference Publication*, 2001.

[120] M. Berges, E. Goldman, H. S. Matthews, L. Soibelman, and K. Anderson, "User-Centered Nonintrusive Electricity Load Monitoring for Residential Buildings," *J. Comput. Civ. Eng.*, 2011.

[121] T. SAITOH, T. OSAKI, R. KONISHI, and K. SUGAHARA, "Current Sensor Based Home Appliance and State of Appliance Recognition," *SICE J. Control. Meas. Syst. Integr.*, 2010.

[122] W. He and Y. Chai, "An Empirical Study on Energy Disaggregation via Deep Learning," in *2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE 2016)*, 2016.

[123] J. Kelly and W. Knottenbelt, "Neural NILM: Deep neural networks applied to energy disaggregation," in *BuildSys 2015 - Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built*, 2015.

[124] K. S. Barsim and B. Yang, "On the Feasibility of Generic Deep Disaggregation for Single-Load Extraction," pp. 1–5, 2018.

[125] C. Zhang, M. Zhong, Z. Wang, N. Goddard, and C. Sutton, "Sequence-to-point learning with neural networks for non-intrusive load monitoring," in *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, 2018.

[126] M. D'Incecco, S. Squartini, and M. Zhong, "Transfer Learning for Non-Intrusive Load Monitoring," *IEEE Trans. Smart Grid*, 2020.

[127] D. Murray, L. Stankovic, V. Stankovic, S. Lulic, and S. Sladojevic, "Transferability of Neural Network Approaches for Low-rate Energy Disaggregation," in *ICASSP, IEEE International Conference on Acoustics,*

Speech and Signal Processing - Proceedings, 2019.

[128] W. Xu, H. Sun, C. Deng, and Y. Tan, "Variational autoencoder for semi-supervised text classification," in *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, 2017.

[129] S. Rezaeifar, O. Taran, and S. Voloshynovskiy, "Classification by re-generation: Towards classification based on variational inference," in *European Signal Processing Conference*, 2018.

[130] J. Bai, S. Kong, and C. Gomes, "Disentangled Variational Autoencoder based Multi-Label Classification with Covariance-Aware Multivariate Probit Model," in *Twenty-Ninth International Joint Conference on Artificial Intelligence*, 2020.

[131] E. Karamatli, A. T. Cemgil, and S. Kirbiz, "Audio Source Separation Using Variational Autoencoders and Weak Class Supervision," *IEEE Signal Process. Lett.*, 2019.

[132] S. Seki, H. Kameoka, L. Li, T. Toda, and K. Takeda, "Underdetermined Source Separation Based on Generalized Multichannel Variational Autoencoder," *IEEE Access*, 2019.

[133] H. Kameoka, L. Li, S. Inoue, and S. Makino, "Semi-Blind Source Separation With Multichannel Variational Autoencoder," *Arxiv*, 2018.

[134] S. Desai, R. Alhadad, A. Mahmood, N. Chilamkurti, and S. Rho, "Multi-state energy classifier to evaluate the performance of the NILM algorithm," *Sensors (Switzerland)*, vol. 19, no. 23, pp. 1–17, 2019.

[135] L. Pereira and N. Nunes, "Performance evaluation in non-intrusive load monitoring: Datasets, metrics, and tools—A review," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. 2018.

[136] J. Z. Kolter and M. J. Johnson, "REDD : A Public Data Set for Energy Disaggregation Research," *SustKDD Work.*, 2011.

[137] J. Kelly and W. Knottenbelt, "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes," *Sci. Data*, 2015.

[138] K. Anderson, A. F. Ocneanu, D. Benitez, D. Carlson, A. Rowe, and M. Bergés, "BLUED : A Fully Labeled Public Dataset for Event-Based Non-Intrusive Load Monitoring Research," in *Proceedings of the 2nd KDD Workshop on Data Mining Applications in Sustainability (SustKDD)*, 2012.

[139] D. Murray, L. Stankovic, and V. Stankovic, "An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study," *Sci. Data*, 2017.

[140] J. Gao, S. Giri, E. C. Kara, and M. Bergés, "PLAID: A public dataset of high-resolution electrical appliance measurements for load identification research," in *BuildSys 2014 - Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, 2014.

[141] S. N. Akshay Uttama Nambi, A. Reyes Lua, and R. Venkatesha Prasad, "LocED: Location-aware Energy Disaggregation Framework," in *BuildSys 2015 - Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built*, 2015.

[142] M. Kahl, A. Haq, and T. Kriechbaumer, "Whited-a worldwide household and industry transient energy data set," *3rd Int. Work. Non-Intrusive Load Monit.*, 2016.

[143] "NILM WIKI." [Online]. Available: http://wiki.nilm.eu/datasets.html.

[144] "irise." [Online]. Available: https://remodece.isr.uc.pt/.

[145] M. Gulati, S. S. Ram, and A. Singh, "An in depth study into using EMI signatures for appliance identification," in *BuildSys 2014 - Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, 2014.

[146] "Demand response market value." [Online]. Available: https://www.smart-energy.com/news/eu-demand-response-market-frost-sullivan/.

[147] A. K. Ehrhardt-martinez and K. a Donnelly, "Advanced Metering Initiatives and Residential Feedback Programs : A Meta-Review for Household Electricity-Saving Opportunities," *Energy*, 2010.

[148] D. Christensen, L. Earle, B. Sparn, D. Christensen, L. Earle, and B. Sparn, "NILM Applications for the Energy-Efficient Home NILM Applications for the Energy-Efficient Home," no. November, 2012.

[149] "Home Energy Management," 2018. [Online]. Available: https://www.marketwatch.com/press-release/home-energy-management-system-hems-global-market-analysis-growth-opportunities-top-key-players-and-forecast-to-2025-2018-08-09.

[150] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE*, 1989.

[151] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Trans. Inf. Theory*, 1967.

[152] roger levy, "Hidden Markov Model inference with the Viterbi algorithm: a mini-example," 2009. [Online]. Available: http://idiom.ucsd.edu/~rlevy/teaching/winter2009/ligncse256/lectures/hmm_viterbi_mini_example.pdf.

[153] N. Henao, K. Agbossou, S. Kelouwani, S. S. Hosseini, and M. Fournier, "Power estimation of multiple two-state loads using a probabilistic non-intrusive approach," *Energies*, 2018.

[154] Z. Ghahramani and M. I. Jordan, "Factorial Hidden Markov Models," *Mach. Learn.*, 1997.

[155] G. Casella and E. I. George, "Explaining the gibbs sampler," *Am. Stat.*, 1992.

[156] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural Networks*, 1989.

[157] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science (80-.).*, 2006.

[158] C. G. Turhan and H. S. Bilge, "Recent Trends in Deep Generative Models: A Review," in *UBMK 2018 - 3rd International Conference on Computer Science and Engineering*, 2018.

[159] R. Salakhutdinov, "Learning Deep Generative Models," *Annu. Rev. Stat. Its Appl.*, 2015.

[160] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 2014.

[161] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Foundations and Trends in Machine Learning*. 2019.

[162] joseph rocca, "Understanding Variational Auto encoders," 2019. [Online]. Available: https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73#:~:text=variational autoencoders (VAEs) are autoencoders,order to ensure a better.

[163] C. C. Aggarwal, *Neural Networks and Deep Learning*. 2018.

[164] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation," in *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, 2013.

[165] J. Wan *et al.*, "A Critical Review of Recurrent Neural Networks for Sequence Learning arXiv : 1506 .

00019v2 [cs . LG] 29 Jun 2015," *Int. J. Comput. Vis.*, 2015.

[166] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural Computation*. 2019.

[167] S. Ryu and S. C. Kim, "Knocking and listening: Learning mechanical impulse response for understanding surface characteristics," *Sensors (Switzerland)*, 2020.

[168] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into Imaging*. 2018.

[169] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*. 2020.

[170] Mohammed Waleed Kadous, "Expanding the Scope of Concept Learning Using Metafeatures," 2002.

[171] H. Yoon, K. Yang, and C. Shahabi, "Feature subset selection and feature ranking for multivariate time series," *IEEE Trans. Knowl. Data Eng.*, 2005.

[172] J. Rosinski, J. Klich, A. Filipkowska, and R. Pettinger, "Top-down and bottom-up approach to competence management implementation: A case of two central banks," 2014.

[173] D. Fischer, T. Wolf, J. Scherer, and B. Wille-Haussmann, "A stochastic bottom-up model for space heating and domestic hot water load profiles for German households," *Energy Build.*, 2016.

[174] J. Widén, A. M. Nilsson, and E. Wäckelgård, "A combined Markov-chain and bottom-up approach to modelling of domestic lighting demand," *Energy Build.*, 2009.

[175] F. L. C. Silva, R. C. Souza, F. L. Cyrino Oliveira, P. M. Lourenco, and R. F. Calili, "A bottom-up methodology for long term electricity consumption forecasting of an industrial sector - Application to pulp and paper sector in Brazil," *Energy*, 2018.

[176] F. L. C. da Silva, F. L. Cyrino Oliveira, and R. C. Souza, "A bottom-up bayesian extension for long term electricity consumption forecasting," *Energy*, 2019.

[177] B. Gao, X. Liu, and Z. Zhu, "A bottom-up model for household load profile based on the consumption behavior of residents," *Energies*, 2018.

[178] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification," *IEEE Trans. Inf. Theory*, 1967.

[179] X. Shi, H. Ming, S. Shakkottai, L. Xie, and J. Yao, "Nonintrusive load monitoring in residential households with low-resolution data," *Appl. Energy*, 2019.

[180] Y. Shou, N. Mamoulis, and D. Cheung, "Fast and exact warping of time series using adaptive segmental approximations," *Mach. Learn.*, 2005.

[181] P.senin, "Dynamic Time Warping Algorithm Review," *Science (80-.).*, 2008.

[182] Elena Tsiporkova, "Dynamic Time Warping algorithm." [Online]. Available: http://www.mathcs.emory.edu/~lxiong/cs730_s13/share/slides/searching_sigkdd2012_DTW.pdf.

APPENDICES

Appendix A

HMM model parameters

Water heater

General model

state	Initial probability
S1	0.21
S2	0.23
S3	0.56

State transition probability	S1	S2	S3
S1	0.51	0.33	0.16
S2	0.53	0.23	0.24
S3	0.03	0.19	0.78

State (emission parameters)	mean	variance
S1	0.5	3
S2	175	101
S3	399	131

Tuned model

House-14

state	Initial probability
S1	0.70
S2	0.15
S3	0.15

State transition probability	S1	S2	S3
S1	0.75	0.2	0.05
S2	0.5	0.05	0.45
S3	0.05	0.15	0.8

State (emission parameters)	mean	variance
S1	2	3
S2	250	25
S3	375	20

House-28

state	Initial probability
S1	0.70
S2	0.1
S3	0.2

State transition probability	S1	S2	S3
S1	0.7	0.2	0.1
S2	0.45	0.1	0.45
S3	0.1	0.1	0.8

State (emission parameters)	mean	variance
S1	2	3
S2	130	100
S3	520	30

House-36

state	Initial probability
S1	0.75
S2	0.1
S3	0.15

State transition probability	S1	S2	S3
S1	0.66	0.33	0.01
S2	0.25	0.3	0.45
S3	0.05	0.1	0.85

State (emission parameters)	mean	variance
S1	3	5
S2	160	120
S3	390	30

Electric heating

General model

Chain 1		Chain 2	
state	Initial probability	state	Initial probability
s1	0.45	s1	0.4
s2	0.3	s2	0.25
s3	0.25	s3	0.35

Chain 1				Chain 2			
Transition probability	S1	S2	S3	Transition probability	S1	S2	S3
S1	0.9	0.05	0.05	S1	0.8	0.05	0.15
S2	0.3	0.6	0.1	S2	0.4	0.55	0.05
S3	0.1	0.1	0.8	S3	0.1	0.15	0.75

Chain1 (emission parameters)			Chain 2 (emission parameters)		
state	mean	variance	state	mean	variance
S1	3	30	S1	8	30
S2	107	101	S2	80	99
S3	195	95	S3	212	100

Tuned model

House 14

Chain 1		Chain 2	
state	Initial probability	state	Initial probability
s1	0.35	s1	0.5
s2	0.35	s2	0.25
s3	0.3	s3	0.25

Chain 1				Chain 2			
Transition prob	S1	S2	S3	Transition prob	S1	S2	S3
S1	0.8	0.1	0.1	S1	0.6	0.15	0.25
S2	0.35	0.5	0.15	S2	0.4	0.4	0.2
S3	0.1	0.2	0.75	S3	0.15	0.1	0.75

Chain1 (emission parameters)			Chain 2 (emission parameters)		
state	mean	variance	state	mean	variance
S1	15	30	S1	162	30
S2	275	101	S2	285	99
S3	700	95	S3	465	100

House 28

Chain 1		Chain 2	
state	Initial probability	state	Initial probability
s1	0.25	s1	0.35
s2	0.45	s2	0.4
s3	0.3	s3	0.25

Chain 1				Chain 2			
Transition prob	S1	S2	S3	Transition prob	S1	S2	S3
S1	0.7	0.25	0.05	S1	0.65	0.15	0.2
S2	0.2	0.5	0.3	S2	0.3	0.4	0.3
S3	0.15	0.25	0.6	S3	0.1	0.1	0.8

Chain1 (emission parameters)			Chain 2 (emission parameters)		
state	mean	variance	state	mean	variance
S1	5	30	S1	37	30
S2	190	101	S2	202	99
S3	317	95	S3	351	100

House 36

Chain 1		Chain 2	
state	Initial probability	state	Initial probability
s1	0.45	s1	0.3
s2	0.4	s2	0.4
s3	0.15	s3	0.3

Chain 1				Chain 2			
Transition prob	S1	S2	S3	Transition prob	S1	S2	S3
S1	0.5	0.35	0.15	S1	0.55	0.15	0.3
S2	0.25	0.4	0.35	S2	0.25	0.45	0.3
S3	0.2	0.35	0.45	S3	0.15	0.2	0.65

Chain1 (emission parameters)			Chain 2 (emission parameters)		
state	mean	variance	state	mean	variance
S1	5	30	S1	95	30
S2	196	101	S2	210	99
S3	420	95	S3	365	100

Washing machine
General model

state	Initial probability
S1	0.45
S2	0.2
S3	0.1
S4	0.25

State transition probability	S1	S2	S3	S4
S1	0.45	0.35	0.15	0.05
S2	0.45	0.15	0.15	0.25
S3	0.5	0.05	0.05	0.4
S4	0.35	0.4	0.1	0.15

State (emission parameters)	mean	variance
S1	4	6
S2	50	38
S3	200	27
S4	300	25

Tuned model

House-14

state	Initial probability
S1	0.4
S2	0.2
S3	0.1
S4	0.3

State transition probability	S1	S2	S3	S4
S1	0.6	0.15	0.15	0.1
S2	0.25	0.65	0.05	0.05
S3	0.01	0.55	0.01	0.43
S4	0.2	0.35	0.1	0.35

State (emission parameters)	mean	variance
S1	8	10
S2	60	40
S3	165	53
S4	335	45

House-28

state	Initial probability
S1	0.4
S2	0.2
S3	0.1
S4	0.3

State transition probability	S1	S2	S3	S4
S1	0.65	0.1	0.15	0.1
S2	0.15	0.65	0.1	0.1
S3	0.01	0.43	0.01	0.55
S4	0.1	0.4	0.4	0.1

State (emission parameters)	mean	variance
S1	8	10
S2	40	20
S3	190	65
S4	379	50

House-36

state	Initial probability
S1	0.3
S2	0.2
S3	0.1
S4	0.4

State transition probability	S1	S2	S3	S4
S1	0.5	0.3	0.15	0.05
S2	0.25	0.4	0.15	0.25
S3	0.01	0.45	0.01	0.53
S4	0.3	0.2	0.1	0.4

State (emission parameters)	mean	variance
S1	5	6
S2	85	20
S3	150	25
S4	310	78

Cloths drier

General model

state	Initial probability
S1	0.45
S2	0.25
S3	0.15
S4	0.15

State transition probability	S1	S2	S3	S4
S1	0.65	0.05	0.2	0.1
S2	0.25	0.65	0.05	0.05
S3	0.1	0.2	0.15	0.55
S4	0.1	0.35	0.45	0.1

State (emission parameters)	mean	variance
S1	4	6
S2	105	40
S3	250	60
S4	364	65

Tuned model

House 14

state	Initial probability
S1	0.4
S2	0.25
S3	0.2
S4	0.15

State transition probability	S1	S2	S3	S4
S1	0.6	0.05	0.3	0.05
S2	0.5	0.3	0.1	0.1
S3	0.1	0.1	0.1	0.7
S4	0.1	0.2	0.1	0.6

State (emission parameters)	mean	variance
S1	3	5
S2	50	35
S3	240	20
S4	375	20

House 28

state	Initial probability
S1	0.5
S2	0.2
S3	0.15
S4	0.15

State transition probability	S1	S2	S3	S4
S1	0.75	0.1	0.1	0.05
S2	0.4	0.1	0.1	0.4
S3	0.1	0.15	0.15	0.6
S4	0.1	0.25	0.1	0.55

State (emission parameters)	mean	variance
S1	8	6
S2	66	20
S3	362	35
S4	443	30

House 36

state	Initial probability
S1	0.45
S2	0.25
S3	0.15
S4	0.15

State transition probability	S1	S2	S3	S4
S1	0.4	0.3	0.2	0.1
S2	0.05	0.05	0.2	0.6
S3	0.1	0.1	0.15	0.65
S4	0.1	0.05	0.1	0.75

State (emission parameters)	mean	variance
S1	3	5
S2	150	70
S3	264	80
S4	400	20

Dish washer

General model

state	Initial probability
S1	0.35
S2	0.3
S3	0.2
S4	0.15

State transition probability	S1	S2	S3	S4
S1	0.5	0.05	0.25	0.2
S2	0.25	0.23	0.3	0.22
S3	0.23	0.16	0.21	0.4
S4	0.35	0.3	0.2	0.15

State (emission parameters)	mean	variance
S1	3	2
S2	75	40
S3	195	85
S4	470	50

House-14

state	Initial probability
S1	0.35
S2	0.35
S3	0.15
S4	0.15

State transition probability	S1	S2	S3	S4
S1	0.6	0.2	0.15	0.05
S2	0.6	0.25	0.1	0.05
S3	0.05	0.4	0.4	0.15
S4	0.1	0.25	0.1	0.55

state	mean	variance
S1	4	2
S2	60	33
S3	265	40
S4	427	85

House-28

state	Initial probability
S1	0.35
S2	0.2
S3	0.2
S4	0.25

State transition probability	S1	S2	S3	S4
S1	0.5	0.25	0.15	0.1
S2	0.33	0.05	0.32	0.3
S3	0.05	0.4	0.1	0.45
S4	0.4	0.05	0.4	0.15

State (emission parameters)	mean	variance
S1	3	3
S2	40	35
S3	190	50
S4	445	50

House-36

state	Initial probability
S1	0.3
S2	0.2
S3	0.25
S4	0.25

State transition probability	S1	S2	S3	S4
S1	0.6	0.05	0.25	0.1
S2	0.35	0.5	0.1	0.05
S3	0.05	0.8	0.05	0.1
S4	0.05	0.65	0.15	0.15

State (emission parameters)	mean	variance
S1	2	4
S2	70	35
S3	130	63
S4	470	50

Variational auto encoder

Water heater

Classifier

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_29 (Conv1D)           (None, 3, 20)             60
_________________________________________________________________
conv1d_30 (Conv1D)           (None, 3, 15)             315
_________________________________________________________________
max_pooling1d_15 (MaxPooling (None, 1, 15)             0
_________________________________________________________________
flatten_15 (Flatten)         (None, 15)                0
_________________________________________________________________
dense_64 (Dense)             (None, 65)                1040
_________________________________________________________________
dense_65 (Dense)             (None, 1)                 66
=================================================================
Total params: 1,481
Trainable params: 1,481
Non-trainable params: 0
_________________________________________________________________
```

Encoder

Layer (type)	Output Shape	Param #	Connected to
encoder_input (InputLayer)	(None, 20)	0	
reshape_13 (Reshape)	(None, 20, 1)	0	encoder_input[0][0]
conv1d_25 (Conv1D)	(None, 17, 55)	275	reshape_13[0][0]
max_pooling1d_13 (MaxPooling1D)	(None, 8, 55)	0	conv1d_25[0][0]
conv1d_26 (Conv1D)	(None, 7, 25)	2775	max_pooling1d_13[0][0]
max_pooling1d_14 (MaxPooling1D)	(None, 3, 25)	0	conv1d_26[0][0]
flatten_13 (Flatten)	(None, 75)	0	max_pooling1d_14[0][0]
dense_55 (Dense)	(None, 450)	34200	flatten_13[0][0]
dense_56 (Dense)	(None, 380)	171380	dense_55[0][0]
dense_57 (Dense)	(None, 180)	68580	dense_56[0][0]
dense_58 (Dense)	(None, 25)	4525	dense_57[0][0]
z_mean (Dense)	(None, 4)	104	dense_58[0][0]
z_log_var (Dense)	(None, 4)	104	dense_58[0][0]
z (Lambda)	(None, 4)	0	z_mean[0][0] z_log_var[0][0]

Total params: 281,943
Trainable params: 281,943
Non-trainable params: 0

Decoder

Layer (type)	Output Shape	Param #
z_sampling (InputLayer)	(None, 4)	0
reshape_14 (Reshape)	(None, 4, 1)	0
conv1d_27 (Conv1D)	(None, 4, 25)	50
up_sampling1d_13 (UpSampling	(None, 4, 25)	0
conv1d_28 (Conv1D)	(None, 4, 35)	910
up_sampling1d_14 (UpSampling	(None, 4, 35)	0
flatten_14 (Flatten)	(None, 140)	0
dense_59 (Dense)	(None, 450)	63450
dense_60 (Dense)	(None, 350)	157850
dense_61 (Dense)	(None, 120)	42120
dense_62 (Dense)	(None, 70)	8470
dense_63 (Dense)	(None, 20)	1420

Total params: 274,270
Trainable params: 274,270
Non-trainable params: 0

Electric heating

Classifier

Layer (type)	Output Shape	Param #
conv1d_35 (Conv1D)	(None, 5, 20)	60
conv1d_36 (Conv1D)	(None, 5, 15)	315
max_pooling1d_18 (MaxPooling	(None, 2, 15)	0
flatten_18 (Flatten)	(None, 30)	0
dense_77 (Dense)	(None, 65)	2015
dense_78 (Dense)	(None, 1)	66

Total params: 2,456
Trainable params: 2,456
Non-trainable params: 0

Encoder

Layer (type)	Output Shape	Param #	Connected to
encoder_input (InputLayer)	(None, 20)	0	
reshape_15 (Reshape)	(None, 20, 1)	0	encoder_input[0][0]
conv1d_31 (Conv1D)	(None, 15, 55)	385	reshape_15[0][0]
max_pooling1d_16 (MaxPooling1D)	(None, 5, 55)	0	conv1d_31[0][0]
conv1d_32 (Conv1D)	(None, 3, 35)	5810	max_pooling1d_16[0][0]
max_pooling1d_17 (MaxPooling1D)	(None, 1, 35)	0	conv1d_32[0][0]
flatten_16 (Flatten)	(None, 35)	0	max_pooling1d_17[0][0]
dense_66 (Dense)	(None, 550)	19800	flatten_16[0][0]
dense_67 (Dense)	(None, 430)	236930	dense_66[0][0]
dense_68 (Dense)	(None, 330)	142230	dense_67[0][0]
dense_69 (Dense)	(None, 240)	79440	dense_68[0][0]
dense_70 (Dense)	(None, 40)	9640	dense_69[0][0]
z_mean (Dense)	(None, 5)	205	dense_70[0][0]
z_log_var (Dense)	(None, 5)	205	dense_70[0][0]
z (Lambda)	(None, 5)	0	z_mean[0][0] z_log_var[0][0]

Total params: 494,645
Trainable params: 494,645
Non-trainable params: 0

Decoder

Layer (type)	Output Shape	Param #
z_sampling (InputLayer)	(None, 5)	0
reshape_16 (Reshape)	(None, 5, 1)	0
conv1d_33 (Conv1D)	(None, 5, 25)	50
up_sampling1d_15 (UpSampling	(None, 5, 25)	0
conv1d_34 (Conv1D)	(None, 5, 35)	910
up_sampling1d_16 (UpSampling	(None, 5, 35)	0
flatten_17 (Flatten)	(None, 175)	0
dense_71 (Dense)	(None, 470)	82720
dense_72 (Dense)	(None, 390)	183690
dense_73 (Dense)	(None, 240)	93840
dense_74 (Dense)	(None, 180)	43380
dense_75 (Dense)	(None, 90)	16290
dense_76 (Dense)	(None, 20)	1820

Total params: 422,700
Trainable params: 422,700
Non-trainable params: 0

Washing machine

Classifier

Layer (type)	Output Shape	Param #
conv1d_41 (Conv1D)	(None, 3, 20)	60
conv1d_42 (Conv1D)	(None, 3, 15)	315
max_pooling1d_21 (MaxPooling	(None, 1, 15)	0
flatten_21 (Flatten)	(None, 15)	0
dense_88 (Dense)	(None, 65)	1040
dense_89 (Dense)	(None, 1)	66

Total params: 1,481
Trainable params: 1,481
Non-trainable params: 0

Encoder

Layer (type)	Output Shape	Param #	Connected to
encoder_input (InputLayer)	(None, 15)	0	
reshape_17 (Reshape)	(None, 15, 1)	0	encoder_input[0][0]
conv1d_37 (Conv1D)	(None, 12, 25)	125	reshape_17[0][0]
max_pooling1d_19 (MaxPooling1D)	(None, 6, 25)	0	conv1d_37[0][0]
conv1d_38 (Conv1D)	(None, 5, 10)	510	max_pooling1d_19[0][0]
max_pooling1d_20 (MaxPooling1D)	(None, 2, 10)	0	conv1d_38[0][0]
flatten_19 (Flatten)	(None, 20)	0	max_pooling1d_20[0][0]
dense_79 (Dense)	(None, 350)	7350	flatten_19[0][0]
dense_80 (Dense)	(None, 280)	98280	dense_79[0][0]
dense_81 (Dense)	(None, 180)	50580	dense_80[0][0]
dense_82 (Dense)	(None, 25)	4525	dense_81[0][0]
z_mean (Dense)	(None, 4)	104	dense_82[0][0]
z_log_var (Dense)	(None, 4)	104	dense_82[0][0]
z (Lambda)	(None, 4)	0	z_mean[0][0] z_log_var[0][0]

Total params: 161,578
Trainable params: 161,578
Non-trainable params: 0

Decoder

Layer (type)	Output Shape	Param #
z_sampling (InputLayer)	(None, 4)	0
reshape_18 (Reshape)	(None, 4, 1)	0
conv1d_39 (Conv1D)	(None, 4, 25)	50
up_sampling1d_17 (UpSampling	(None, 4, 25)	0
conv1d_40 (Conv1D)	(None, 4, 35)	910
up_sampling1d_18 (UpSampling	(None, 4, 35)	0
flatten_20 (Flatten)	(None, 140)	0
dense_83 (Dense)	(None, 350)	49350
dense_84 (Dense)	(None, 250)	87750
dense_85 (Dense)	(None, 120)	30120
dense_86 (Dense)	(None, 70)	8470
dense_87 (Dense)	(None, 15)	1065

Total params: 177,715
Trainable params: 177,715
Non-trainable params: 0

Dish washer

Classifier

Layer (type)	Output Shape	Param #
conv1d_47 (Conv1D)	(None, 3, 15)	45
conv1d_48 (Conv1D)	(None, 3, 10)	160
max_pooling1d_24 (MaxPooling	(None, 1, 10)	0
flatten_24 (Flatten)	(None, 10)	0
dense_99 (Dense)	(None, 35)	385
dense_100 (Dense)	(None, 1)	36

Total params: 626
Trainable params: 626
Non-trainable params: 0

Encoder

Layer (type)	Output Shape	Param #	Connected to
encoder_input (InputLayer)	(None, 15)	0	
reshape_19 (Reshape)	(None, 15, 1)	0	encoder_input[0][0]
conv1d_43 (Conv1D)	(None, 12, 25)	125	reshape_19[0][0]
max_pooling1d_22 (MaxPooling1D)	(None, 6, 25)	0	conv1d_43[0][0]
conv1d_44 (Conv1D)	(None, 5, 10)	510	max_pooling1d_22[0][0]
max_pooling1d_23 (MaxPooling1D)	(None, 2, 10)	0	conv1d_44[0][0]
flatten_22 (Flatten)	(None, 20)	0	max_pooling1d_23[0][0]
dense_90 (Dense)	(None, 350)	7350	flatten_22[0][0]
dense_91 (Dense)	(None, 280)	98280	dense_90[0][0]
dense_92 (Dense)	(None, 180)	50580	dense_91[0][0]
dense_93 (Dense)	(None, 25)	4525	dense_92[0][0]
z_mean (Dense)	(None, 4)	104	dense_93[0][0]
z_log_var (Dense)	(None, 4)	104	dense_93[0][0]
z (Lambda)	(None, 4)	0	z_mean[0][0] z_log_var[0][0]

Total params: 161,578
Trainable params: 161,578
Non-trainable params: 0

Decoder

Layer (type)	Output Shape	Param #
z_sampling (InputLayer)	(None, 4)	0
reshape_20 (Reshape)	(None, 4, 1)	0
conv1d_45 (Conv1D)	(None, 4, 25)	50
up_sampling1d_19 (UpSampling	(None, 4, 25)	0
conv1d_46 (Conv1D)	(None, 4, 35)	910
up_sampling1d_20 (UpSampling	(None, 4, 35)	0
flatten_23 (Flatten)	(None, 140)	0
dense_94 (Dense)	(None, 370)	52170
dense_95 (Dense)	(None, 220)	81620
dense_96 (Dense)	(None, 110)	24310
dense_97 (Dense)	(None, 60)	6660
dense_98 (Dense)	(None, 15)	915

Total params: 166,635
Trainable params: 166,635
Non-trainable params: 0

Clothes drier

Classifier

Layer (type)	Output Shape	Param #
conv1d_53 (Conv1D)	(None, 3, 20)	60
conv1d_54 (Conv1D)	(None, 3, 15)	315
max_pooling1d_27 (MaxPooling	(None, 1, 15)	0
flatten_27 (Flatten)	(None, 15)	0
dense_110 (Dense)	(None, 65)	1040
dense_111 (Dense)	(None, 1)	66

Total params: 1,481
Trainable params: 1,481
Non-trainable params: 0

Encoder

```
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
encoder_input (InputLayer)      (None, 15)           0

reshape_21 (Reshape)            (None, 15, 1)        0           encoder_input[0][0]

conv1d_49 (Conv1D)              (None, 12, 55)       275         reshape_21[0][0]

max_pooling1d_25 (MaxPooling1D) (None, 6, 55)        0           conv1d_49[0][0]

conv1d_50 (Conv1D)              (None, 5, 25)        2775        max_pooling1d_25[0][0]

max_pooling1d_26 (MaxPooling1D) (None, 2, 25)        0           conv1d_50[0][0]

flatten_25 (Flatten)            (None, 50)           0           max_pooling1d_26[0][0]

dense_101 (Dense)               (None, 450)          22950       flatten_25[0][0]

dense_102 (Dense)               (None, 380)          171380      dense_101[0][0]

dense_103 (Dense)               (None, 220)          83820       dense_102[0][0]

dense_104 (Dense)               (None, 25)           5525        dense_103[0][0]

z_mean (Dense)                  (None, 4)            104         dense_104[0][0]

z_log_var (Dense)               (None, 4)            104         dense_104[0][0]

z (Lambda)                      (None, 4)            0           z_mean[0][0]
                                                                 z_log_var[0][0]
==================================================================================================
Total params: 286,933
Trainable params: 286,933
Non-trainable params: 0
```

Decoder

```
Layer (type)                 Output Shape              Param #
=================================================================
z_sampling (InputLayer)      (None, 4)                 0

reshape_22 (Reshape)         (None, 4, 1)              0

conv1d_51 (Conv1D)           (None, 4, 25)             50

up_sampling1d_21 (UpSampling (None, 4, 25)             0

conv1d_52 (Conv1D)           (None, 4, 35)             910

up_sampling1d_22 (UpSampling (None, 4, 35)             0

flatten_26 (Flatten)         (None, 140)               0

dense_105 (Dense)            (None, 450)               63450

dense_106 (Dense)            (None, 350)               157850

dense_107 (Dense)            (None, 140)               49140

dense_108 (Dense)            (None, 70)                9870

dense_109 (Dense)            (None, 15)                1065
=================================================================
Total params: 282,335
Trainable params: 282,335
Non-trainable params: 0
```

Statistical Rules based method

Water heater

Max value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0	8	0.053583	15	0.07295
2	0.018076	9	0.057456	16	0.07295
3	0.060039	10	0.054229	17	0.060684
4	0.051646	11	0.058748	18	0.036152
5	0.063267	12	0.065849	19	0.00581
6	0.052292	13	0.078115	20	0
7	0.06133	14	0.076824		

Min value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0.082287	8	0.037486	15	0.042424
2	0.069182	9	0.037364	16	0.045898
3	0.060892	10	0.03706	17	0.051201
4	0.053273	11	0.036877	18	0.059125
5	0.0473	12	0.037974	19	0.060892
6	0.042667	13	0.037974	20	0.082287
7	0.039193	14	0.038644	15	0.042424

Energy difference

position	**Energy difference mean**	**position**	**Energy difference mean**
1	-3.95E-01	11	-5.85E-03
2	-3.08E-01	12	-2.01E-02
3	-1.95E-01	13	-2.61E-02
4	-1.35E-01	14	-5.89E-02
5	-7.45E-02	15	-1.03E-01
6	-4.02E-02	16	-1.43E-01
7	-1.51E-04	17	-2.03E-01
8	4.64E-03	18	-2.70E-01
9	6.47E-03	19	-3.40E-01
10	0.00E+00	20	-3.95E-01

Position	Energy difference standard deviation	Position	Energy difference standard deviation
1	3.44E-01	11	2.46E-01
2	4.20E-01	12	3.42E-01
3	4.94E-01	13	3.95E-01
4	5.01E-01	14	4.31E-01
5	4.94E-01	15	0.453191
6	4.57E-01	16	0.472661
7	4.14E-01	17	0.46096
8	3.53E-01	18	0.434906
9	2.54E-01	19	0.370924
10	2.50E-01	20	0.344268

Gradient

Position	Gradient mean	Position	Gradient mean
2	0.087482	12	-0.01425
3	0.112528	13	-0.006
4	0.060724	14	-0.03277
5	0.060256	15	-0.04381
6	0.034229	16	-0.04048
7	0.04008	17	-0.06018
8	0.004796	18	-0.06657
9	0.00183	19	-0.07033
10	-0.00647	20	-0.05521
11	-0.00585		

Position	Gradient standard deviation	Position	Gradient standard deviation
2	0.234683	12	0.24721
3	0.296011	13	0.250583
4	0.288952	14	0.267741
5	0.31313	15	0.266416
6	0.301926	16	0.269329
7	0.297402	17	0.256874
8	0.269858	18	0.234441
9	0.259949	19	0.223468
10	0.253942	20	0.143124
11	0.246288		

Day of week

Day of week	P (dow)	Day of week	P (dow)
1	0.144597	5	0.138976
2	0.145846	6	0.134603
3	0.142723	7	0.147408
4	0.145846		

Time of use

hour	P(h)	hour	P(h)	hour	P(h)
1	0.059963	9	0.00406	17	0.045909
2	0.042473	10	0.005621	18	0.012492
3	0.024984	11	0.005621	19	0.03123
4	0.022174	12	0.004372	20	0.069956
5	0.022798	13	0.004372	21	0.13148
6	0.019988	14	0.049344	22	0.12742
7	0.006558	15	0.049032	23	0.119613
8	0.003748	16	0.050281	24	0.086508

Gradient rate

Position	Gradient rate mean value	Position	Gradient rate mean value
2	0.16	12	-265.69
3	-101.92	13	-263.58
4	-124.30	14	-255.50
5	-185.54	15	-341.31
6	-206.17	16	-356.91
7	-234.45	17	-387.03
8	-221.70	18	-403.65
9	-234.34	19	-391.67
10	-240.04	20	-552.13
11	-237.84		

Position	Gradient rate standard deviation	Position	Gradient rate standard deviation
2	3.53E-01	12	1.25E+03
3	6.22E+02	13	1.28E+03
4	7.61E+02	14	1.26E+03
5	1.00E+03	15	1.33E+03
6	1.03E+03	16	1.36E+03
7	1.14E+03	17	1.41E+03
8	1.16E+03	18	1.39E+03
9	1.14E+03	19	1.47E+03
10	1.22E+03	20	1.43E+03
11	1.20E+03		

Laplacian

Position	Laplacian mean	Position	Laplacian mean
3	0.025046	12	-0.00839
4	-0.0518	13	0.008247
5	-0.00047	14	-0.02676
6	-0.02603	15	-0.01104
7	0.005851	16	0.003322
8	-0.03528	17	-0.01969
9	-0.00297	18	-0.00639
10	-0.00831	19	-0.00376
11	0.000621	20	0.015115

Position	Laplacian Standard Deviation	Position	Laplacian Standard Deviation
3	0.391066	12	0.354164
4	0.446196	13	0.348833
5	0.429094	14	0.373334
6	0.451573	15	0.37292
7	0.417443	16	0.372286
8	0.399741	17	0.382361
9	0.373729	18	0.349569
10	0.370893	19	0.338605
11	0.352602	20	0.25151

Electric heater

Time of use

hour	P(h)	hour	P(h)	hour	P(h)
1	0.038848	9	0.048014	17	0.04234
2	0.040594	10	0.044086	18	0.044959
3	0.040157	11	0.040157	19	0.051506
4	0.035792	12	0.034046	20	0.046268
5	0.04234	13	0.031427	21	0.041903
6	0.047141	14	0.031864	22	0.044522
7	0.051942	15	0.029681	23	0.041467
8	0.052815	16	0.035792	24	0.04234

Max value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0.104828	6	0.053793	11	0.06069
2	0.070345	7	0.052414	12	0.06069
3	0.064828	8	0.055172	13	0.068966
4	0.053793	9	0.057931	14	0.08
5	0.053793	10	0.06069	15	0.102069

Min value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0.080838	6	0.062447	11	0.062019
2	0.074423	7	0.061591	12	0.062874
3	0.071001	8	0.062019	13	0.066296
4	0.067151	9	0.059453	14	0.068435
5	0.065868	10	0.061163	15	0.074423

Energy difference

Position	Energy difference mean	Position	Energy difference mean
	-0.00346	9	0.004419
2	-0.00589	10	0.007898
3	-0.00832	11	0.009625
4	-0.00852	12	0.011274
5	-0.00928	13	0.011532
6	-0.00595	14	0.014445
7	-0.00154	15	0.017169
8	0		

Position	Energy difference Standard deviation	Position	Energy difference Standard deviation
1	0.52265	9	0.305338
2	0.500517	10	0.36945
3	0.448685	11	0.375782
4	0.447364	12	0.455138
5	0.374009	13	0.464718
6	0.362367	14	0.515622
7	0.305771	15	0.545348
8	0.02		

Gradient

Position	Gradient mean value	Position	Gradient mean value
1	-0.00243	8	0.004419
2	-0.00243	9	0.003479
3	-0.0002	10	0.001726
4	-0.00076	11	0.00165
5	0.003328	12	0.000257
6	0.004409	13	0.002913
7	0.001539	14	0.002724

Position	Gradient Standard Deviation	Position	Gradient Standard Deviation
1	0.308152	8	0.305338
2	0.302578	9	0.306952
3	0.300193	10	0.309105
4	0.297315	11	0.311237
5	0.303931	12	0.311698
6	0.305391	13	0.315074
7	0.305771	14	0.321357

Gradient rate

Position	Gradient rate Mean value	Position	Gradient rate Mean Value
1	-257.213	8	-265.564
2	-279.599	9	-276.061
3	-264.024	10	-294.628
4	-258.522	11	-276.426
5	-266.435	12	-275.79
6	-269.62	13	-275.83
7	-272.518	14	-284.294

Position	Gradient Rate Standard Deviation	Position	Gradient Rate Standard Deviation
1	932.1523	8	977.569
2	959.6207	9	994.8818
3	951.7981	10	1023.849
4	923.2562	11	1000.309
5	929.006	12	999.3079
6	947.1179	13	983.8938
7	966.8452	14	1009.211

Day of week

Day of week	P(dow)	Day of week	P(dow)
1	0.134003	5	0.152335
2	0.144915	6	0.139677
3	0.150153	7	0.13182
4	0.147097		

Laplacian

Position	Laplacian Mean value	Position	Laplacian Mean Value
1	-1.25E-06	8	-9.40E-04
2	2.24E-03	9	-1.75E-03
3	-5.62E-04	10	-7.67E-05
4	4.08E-03	11	-1.39E-03
5	1.08E-03	12	2.66E-03
6	-2.87E-03	13	-1.89E-04
7	2.88E-03		

Position	Laplacian Standard Deviation	Position	Laplacian Standard Deviation
1	0.486339	8	0.483551
2	0.479243	9	0.487913
3	0.476005	10	0.492431
4	0.477427	11	0.490978
5	0.48506	12	0.490793
6	0.486757	13	0.499274
7	0.483822		

Washing machine

Time of use

hour	P(h)	hour	P(h)	hour	P(h)
1	0.00504	9	0.042339	17	0.045363
2	0.002016	10	0.075605	18	0.043347
3	0	11	0.053427	19	0.061492
4	0	12	0.054435	20	0.086694
5	0.001008	13	0.044355	21	0.097782
6	0.002016	14	0.061492	22	0.086694
7	0.019153	15	0.064516	23	0.045363
8	0.042339	16	0.051411	24	0.014113

Day of week

Day of week	P(dow)	Day of week	P(dow)
1	0.136089	5	0.108871
2	0.134073	6	0.15121
3	0.142137	7	0.193548
4	0.134073		

Energy difference

Position	Energy difference Mean value	Position	Energy difference Mean value
1	-0.22747	9	-0.04561
2	-0.11746	10	-0.05048
3	0.041161	11	-0.11678
4	0.105079	12	-0.16219
5	0.111377	13	-0.17634
6	0.116343	14	-0.19978
7	0.072898	15	-0.22747
8	0		

Position	Energy difference Standard Deviation	Position	Energy difference Standard Deviation
1	0.283282	9	0.357803
2	0.429124	10	0.370793
3	0.551904	11	0.325185
4	0.539662	12	0.288885
5	0.495747	13	0.295184
6	0.473971	14	0.275573
7	0.386819	15	0.283282
8	0.02		

Gradient

Position	Gradient Mean value	Position	Gradient Mean value
1	0.110008	9	-0.00487
2	0.158619	10	-0.06631
3	0.063918	11	-0.04541
4	0.006298	12	-0.01415
5	0.004966	13	-0.02344
6	-0.04344	14	-0.02768
7	-0.0729		
8	-0.04561		

Position	Gradient Standard Deviation	Position	Gradient Standard Deviation
1	0.258149	9	0.321908
2	0.345669	10	0.240923
3	0.451836	11	0.166972
4	0.478081	12	0.088106
5	0.47161	13	0.075553
6	0.429476	14	0.059056
7	0.386819		
8	0.357803		

Gradient rate

Position	Gradient Rate Mean value	Position	Gradient Rate Mean value
1	0.21	9	-62.21
2	0.01	10	-144.78
3	-1.36	11	-199.16
4	-1.95	12	-205.87
5	-1.95	13	-257.87
6	-21.26	14	-276.81
7	-32.85		
8	-44.56		

Position	Gradient Rate Standard Deviation	Position	Gradient Rate Standard Deviation
1	3.97E-01	9	2.95E+02
2	1.66E+00	10	4.71E+02
3	8.50E+00	11	5.34E+02
4	5.48E+00	12	5.41E+02
5	5.60E+00	13	5.80E+02
6	1.63E+02	14	5.91E+02
7	2.24E+02		
8	2.54E+02		

Laplacian

Position	Laplacian Mean value	Position	Laplacian Mean value
1	0.048611	8	0.040742
2	-0.0947	9	-0.06144
3	-0.05762	10	0.020899
4	-0.00133	11	0.031256
5	-0.04841	12	-0.00929
6	-0.02945	13	-0.00424
7	0.027289		

Position	Laplacian Standard Deviation	Position	Laplacian Standard Deviation
1	0.452085	8	0.561201
2	0.645184	9	0.463504
3	0.714009	10	0.336872
4	0.724712	11	0.194338
5	0.711748	12	0.133225
6	0.661089	13	0.116622
7	0.600203		

Max value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0	6	0.153333	11	0.031667
2	0.043333	7	0.123333	12	0
3	0.156667	8	0.065	13	0
4	0.176667	9	0.046667	14	0
5	0.165	10	0.038333	15	0

Min value position

Position	P (pos)	Position	P(pos)	Position	P (pos)
1	0.14437	6	0.025987	11	0.040183
2	0.114293	7	0.022137	12	0.059913
3	0.0859	8	0.021415	13	0.080125
4	0.063523	9	0.021655	14	0.10948
5	0.041627	10	0.025024	15	0.14437

Cloths dryer

Time of use

hour	P(h)	hour	P(h)	hour	P(h)
1	0.009259	9	0.060185	17	0.043981
2	0.011574	10	0.05787	18	0.053241
3	0.009259	11	0.048611	19	0.05787
4	0.002315	12	0.071759	20	0.069444
5	0.006944	13	0.071759	21	0.050926
6	0.034722	14	0.076389	22	0.041667
7	0.020833	15	0.064815	23	0.023148
8	0.046296	16	0.060185	24	0.006944

Day of week

Day of week	P (dow)	Day of week	P (dow)
1	0.168981	5	0.083333
2	0.092593	6	0.122685
3	0.145833	7	0.236111
4	0.150463		

Max value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0	6	0.130564	11	0.060831
2	0.008902	7	0.109792	12	0.025223
3	0.087537	8	0.111276	13	0.007418
4	0.149852	9	0.094955	14	0.001484
5	0.137982	10	0.074184	15	0

Min value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0.143369	6	0.032736	11	0.043489
2	0.106332	7	0.025806	12	0.058303
3	0.078853	8	0.021983	13	0.078375
4	0.059259	9	0.025568	14	0.105854
5	0.044206	10	0.032497	15	0.143369

Energy difference

Position	Energy Difference Mean	Position	Energy Difference mean
1	-0.57157	9	-0.06303
2	-0.44483	10	-0.14619
3	-0.22041	11	-0.24195
4	-0.08438	12	-0.35429
5	-0.0058	13	-0.48645
6	0.028329	14	-0.55524
7	0.027351	15	-0.57157
8	0		

Position	Energy Difference Standard Deviation	Position	Energy Difference Standard Deviation
1	0.379151	9	0.286634
2	0.486258	10	0.413654
3	0.59832	11	0.472531
4	0.606113	12	0.4729
5	0.552172	13	0.425045
6	0.457043	14	0.390111
7	0.299848	15	0.379151
8	0.05		

Gradient

Position	Gradient Mean value	Position	Gradient Mean value
1	0.12674	8	-0.06303
2	0.224416	9	-0.08316
3	0.136031	10	-0.09576
4	0.07858	11	-0.11234
5	0.034128	12	-0.13216
6	-0.00098	13	-0.06879
7	-0.02735	14	-0.01633

Position	Gradient Standard Deviation	Position	Gradient Standard Deviation
1	0.257402	8	0.286634
2	0.319056	9	0.278591
3	0.311599	10	0.262037
4	0.311563	11	0.23097
5	0.311265	12	0.229384
6	0.313147	13	0.152665
7	0.299848	14	0.067076

Gradient Rate

Position	Gradient Rate Mean value	Position	Gradient Rate Mean value
1	0.258027	8	-77.4121
2	0.311038	9	-80.9664
3	-21.5107	10	-91.1461
4	-38.3021	11	-99.2781
5	-41.6198	12	-111.423
6	-49.295	13	-143.845
7	-52.704	14	-163.276

Position	Gradient Rate Standard Deviation	Position	Gradient Rate Standard Deviation
1	4.23E-01	8	5.96E+02
2	4.41E-01	9	5.98E+02
3	2.45E+02	10	6.08E+02
4	3.31E+02	11	6.11E+02
5	3.56E+02	12	6.20E+02
6	3.82E+02	13	6.62E+02
7	3.91E+02	14	6.71E+02

Laplacian

Position	Laplacian Mean Value	Position	Laplacian Mean value
1	0.097676	8	-0.02014
2	-0.08838	9	-0.0126
3	-0.05745	10	-0.01658
4	-0.04445	11	-0.01981
5	-0.03511	12	0.063362
6	-0.02637	13	0.052466
7	-0.03567		

Position	Laplacian Standard Deviation	Position	Laplacian Standard Deviation
1	0.397859	8	0.383337
2	0.469747	9	0.361136
3	0.434453	10	0.344996
4	0.412647	11	0.32475
5	0.408523	12	0.25306
6	0.407614	13	0.149529
7	0.392425		

Dishwasher

Time of use

hour	P(h)	hour	P(h)	hour	P(h)
1	0.026764	9	0.036496	17	0.014599
2	0.007299	10	0.029197	18	0.014599
3	0.002433	11	0.012165	19	0.053528
4	0.002433	12	0.051095	20	0.111922
5	0	13	0.090024	21	0.10219
6	0.012165	14	0.077859	22	0.107056
7	0.03163	15	0.068127	23	0.055961
8	0.012165	16	0.026764	24	0.053528

Energy Difference

Position	Energy Difference Mean	Position	Energy Difference Mean
1	-0.40879	9	-0.04292
2	-0.33232	10	-0.10143
3	-0.19806	11	-0.139
4	-0.10943	12	-0.2604
5	-0.05724	13	-0.3651
6	-0.00984	14	-0.39429
7	-0.00115	15	-0.40879
8	0		

Position	Energy Difference Standard Deviation	Position	Energy Difference Standard Deviation
1	0.364396	9	0.49684
2	0.421209	10	0.55549
3	0.455146	11	0.482015
4	0.511265	12	0.451721
5	0.55644	13	0.395219
6	0.634262	14	0.373367
7	0.528224	15	0.364396
8	0.02		

Gradient

Position	Gradient Mean	position	Gradient Mean
1	0.076469	8	-0.04292
2	0.134262	9	-0.05851
3	0.088627	10	-0.03757
4	0.052196	11	-0.12141
5	0.047393	12	-0.10469
6	0.008692	13	-0.02919
7	0.001151	14	-0.01451

Position	Gradient Standard Deviation	Position	Gradient Standard Deviation
1	0.213296	8	0.49684
2	0.304238	9	0.476299
3	0.370934	10	0.445807
4	0.455931	11	0.33725
5	0.48137	12	0.251167
6	0.511628	13	0.107155
7	0.528224	14	0.062955

Gradient Rate

Position	Gradient Rate Mean	Position	Gradient Rate Mean
1	0.169541	8	-91.4161
2	0.185954	9	-114.568
3	-4.25921	10	-126.133
4	-4.77001	11	-132.281
5	-20.9492	12	-132.438
6	-46.3882	13	-129.882
7	-49.685	14	-145.057

Position	Gradient Rate Standard Deviation	Position	Gradient Rate Standard Deviation
1	3.71E-01	8	5.87E+02
2	7.27E-01	9	5.99E+02
3	5.78E+01	10	5.07E+02
4	5.82E+01	11	5.14E+02
5	2.49E+02	12	5.14E+02
6	3.97E+02	13	5.15E+02
7	4.01E+02	14	6.30E+02

Laplacian

Position	Laplacian Mean	Position	Laplacian Mean
1	0.057793	8	-0.01559
2	-0.04564	9	0.020939
3	-0.03643	10	-0.08384
4	-0.0048	11	0.016713
5	-0.0387	12	0.075504
6	-0.00754	13	0.014683
7	-0.04407		

Position	Laplacian Standard Deviation	Position	Laplacian Standard Deviation
1	0.395967	8	0.790864
2	0.532401	9	0.744608
3	0.630606	10	0.650282
4	0.724147	11	0.48917
5	0.781463	12	0.281136
6	0.815041	13	0.122258
7	0.817961		

Day of week

Day of week	P(dow)	Day of week	P(dow)
1	0.116788	5	0.143552
2	0.121655	6	0.175182
3	0.153285	7	0.180049
4	0.109489		

Max value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0	6	0.131579	11	0.088816
2	0.011513	7	0.126645	12	0.034539
3	0.065789	8	0.121711	13	0.004934
4	0.092105	9	0.120066	14	0.001645
5	0.108553	10	0.092105	15	0

Min value position

Position	P(pos)	Position	P(pos)	Position	P(pos)
1	0.137489	6	0.027039	11	0.044225
2	0.114115	7	0.014895	12	0.066911
3	0.090972	8	0.009166	13	0.089826
4	0.068057	9	0.014665	14	0.113428
5	0.045371	10	0.026352	15	0.137489