10-2020

Energy Disaggregation Using Multilabel Binarization and Gaussian Naive Bayes Classifier

Patrick Adjei

Nittin Sethi

Camila de Souza

Miriam A M Capretz

Energy Disaggregation using Multilabel Binarization and Gaussian Naive Bayes Classifier

Patrick Adjei*, Nittin S. Sethi†, Camila P. E. de Souza‡, Miriam A. M. Capretz§
* † §*Department of Electrical and Computer Engineering,* ‡*Department of Statistical and Actuarial Sciences*
Western University, London, Ontario, Canada, N6A 5B9
Email: *padjei@uwo.ca,†nsethi8@uwo.ca, ‡camila.souza@uwo.ca §mcapretz@uwo.ca

***Abstract*—Nearly everywhere across the globe, energy demand is increasing, as new residential buildings emerge and population growth continues. It is important to meet the energy demand while still generating less energy to make up for what is needed and energy disaggregation can assist in this process. In this work, we propose an energy disaggregation method that would allow consumers to be informed on the state (on or off) of their appliances at any point in time. This would enable them to adjust their appliance usage thereby consuming less energy. Our disaggregation approach can disaggregate any combination of appliances among a set of appliances. This is done by creating and fitting *n*-models for the *n*-combinations possible in a strategic manner. Prior to the models, Multilabel Binarization has been used to deduce the labels of the data, then the models are developed and fitted using Gaussian Naive Bayesian Classifier. Thereafter, with Jaccard Coefficient per appliance and accuracy, recall, precision and f1-score were used for overall score. We evaluate these models with the original and noisy synthesized test data, resulting in an average overall F1-score of 91% for the original and 87.8% for the synthesized test data.**

***Keywords*—Non-Intrusive Load Monitoring, Energy Disaggregation, Multilabel Binarization, Gaussian Naive Bayes Classifier.**

I. Introduction

Around 78% of the greenhouse gas emission is due to the production and consumption of electrical energy [1]. Akhmat et al. [2] illustrates that strategic energy consumption can reduce global warming which implies reducing greenhouse gases. One solution that could lead to reduce energy consumption is with energy disaggregation, which is itemizing the consumptions of home appliances. This can give information about the state of each appliances and could assist in using appliances more efficiently [3], [4]. There are two approaches to energy disaggregation. The Intrusive Load Monitoring (ILM) approach and the Non-Intrusive Load Monitoring approach [5]. The ILM approach requires the installation of measuring devices for recording the property of appliances of interest. The NILM approach records the property of interested appliances from a single point of reading. Usually, the recording is done at the house meter. These two approaches are complex in their own way. The ILM complexity is within the building where the connectivity and installation of the measuring devices occur. NILM complexity lies within the software that is to itemize each appliance from a single reading. However, there is a trade-off in practicality and measurement accuracy [6]. On a large scale for ILM, it is impractical to install measuring units for appliances for many houses. On the other hand, a direct reading of the appliance results in a more accurate number. With NILM, it is more practical as there are not nearly as many measuring unit installations for energy recording. The downfall is that the disaggregation from the single point of measure is simply an approximation the software makes.

Many residences have common appliances, a house with a maximum number of distinct appliances would be a superset of houses with less distinct appliances. Hypothetically, it is best to gather many sets of houses that contain the same distinct appliances. Armel et al. [7] describe the problem of few distinct homes which includes distinct appliances. Ideally with different manufacturers and manufacturing date, the variance in appliance profile could be captured to strengthen an energy disaggregation model. This paper uses an autoencoder network as a synthesizing method for the testing data to introduce adequate noise which tries to simulate the energy consumption of the same appliances of different manufacturers.

In this paper we propose a new approach to disaggregate energy consumption of houses in a per minute basis. With low sampling rates, it is ideal to depict appliances as on or off [6]. Hence we treat the appliances as type 1 loads (profiling the appliances with On/Off states) [8]. The paper uses Multilabel Binarizer to generate a map with the state combinations (on or off) of the appliances. The indexes of this map are used as labels for the training data. A Gaussian Naive Bayes Classifier is created using the training data. The output of the model is used to get the corresponding state combinations of the appliances from the map thereby disaggregating the total energy consumption of a house. The advantage of this approach is that it creates energy disaggregation models for different combinations of appliances. Addressed differently by Kelly and Knottenbelt [9], they synthesized data to address the energy consumption for the missing appliances in specific houses which have a subset of the total appliances. Using our approach we have models for different appliance combinations, which eliminates the need to synthesize training data for the houses that do not have every appliance.

This paper is organized as follows: *section II* discusses related work on energy disaggregation. *Section III* explains the methodology used to achieve the proposed models. *Section IV* will explain our synthesizing approach for the testing data, the evaluation process and reveals the results obtained from the models. It is followed by *section V* which provides the conclusion and future work.

II. Related Work

In this section we describe some of the work done to resolve the problem of energy disaggregation. In papers such as [10], [11], [12], [13], [14], [15], [16], [17] and [9], use deep learning to disaggregate energy consumption. To address scalability for models to generalize for many different homes, or variants of an appliance, Barsim and Yang [10] developed a generic deep Convolutional Neural Network (CNN) with fixed hyper-parameters and evaluated it on a variety of appliances. Kaselimi et al. [11] proposed an approach for progressive model updates that can effectively adapt to a wider and more diverse range of instances and conditions. Additionally, Murray et al. [16] presented two different neural network architectures, one, based on convolutional neural network, and another based on gated recurrent unit to scale NILM. However, in our proposed approached, we are building multiple models for the different appliance combinations instead of one generic model. This scales better to multiple houses as each house will have a different set of appliances.

For comparing different NILM methods and data requirements, Çavdar and Faryad [12] used convolutional neural network, 1D CNN-RNN, and long short-term memory which they managed to disaggregate accurately using their proposed architecture. Schirmer and Mporas [14] used K-Nearest-Neighbours, Support Vector Machines, Deep Neural Networks and Random Forest algorithms were evaluated across five datasets. From this, they demonstrated the importance of selecting both appropriate features and regression algorithms. To improve the traditional dictionary learning, Khodayar et al. [13] proposed a novel optimization program where the dictionary and its sparse coefficients are optimized simultaneously with a deep neural model to extract powerful nonlinear features from the energy signals. Shin et al. [15] applied three classification algorithms: vanilla Deep Neural Network, Machine Learning with feature engineering, and CNN with hyper-parameter tuning along with Subtask Gated Network to the dataset. From this, they concluded that lower sampling rate in data performs significantly worse. A newly non-extensive data approach that was used by Schirmer et al. [18] is elastic matching. In this approach they used aggregated energy reading frames to compare against reference frames. From this, they used an elastic matching algorithm to acquire a function that maps to the reference frame. From the reference frame they could determine the appliances that makeup the reference signal. [12], [14], [13], [15] and [18] are not concerned with scaling energy disaggregation to multiple houses whereas our proposed approach is.

Kelly and Knottenbelt [9] considered different subset of appliances in homes, by synthesizing the data before model training. This inspired us to synthesize data using auto-encoder network for the testing data. More familiarly, in image classification, to generate more data, rotation, transpose and more complex image processing is performed on the image. More recently, auxiliary classifying generative adversarial network was used to predict images as a form of synthesizing, introduced by Odena et al. [19] for a more resilient model. In energy disaggregation, synthesizing could be done by randomly combining appliance activation. For example, randomly combining washer and fridge for homes that only have these two appliances from the super-set [9]. These activations are the power drawn from an appliance over a complete cycle of that appliance. With that, among their appliances of interest, they made it so there is a 50% chance for the appliance signature to appear in the training sequence, they were able to train a model with both synthetic aggregated data and the actual aggregated data to generalize better [9]. However, the synthesization for the testing data used in our work is different as auto-encoder network is used instead of taking random activations. Additionally, the combination of appliances activation is ideal if the model creation consists solely of the aggregated appliances being recorded from the meter. On residential homes, it is likely that the aggregated data include more appliances and not only those of interest, unless the disaggregation is for every single appliance in the house. This means, during training the model would not take into consideration the different residuals possible for distinct houses. Synthesizing the data should consider the residuals when randomly selecting appliances activation.

Elhamifar and Sastry [20], addressed the problem of energy disaggregation by learning energy powerlets (the energy snippets which represent the signature consumption patterns of a device) from the different appliances. Firstly, a dictionary of the powerlets of various appliances is made from the training data. Once the dictionary is prepared, the various combination of powerlets is tested to match the aggregate signal. However, the combination of powerlets is not unique so they performed disaggregation optimization to find the optimum combination of powerlets. Disaggregation optimization is contingent upon a few constraints like device sparsity, information of the devices which are generally used together or not as well as temporal consistency. The drawback of their approach [20] is that it is not efficient in disaggregating the consumption of the same appliance which are made by different companies or manufacturers and hence will not perform well on new homes which have the same appliance but from different manufacturers. However, it encouraged us to make a dictionary of the state combination of appliances and use it for label generation.

III. Methodology

In this section, our proposed energy disaggregation approach is discussed in detail. *Subsection A* expands on the dataset and the type of input used followed by *subsection B* which illustrates the overview of the approach. *Subsection C* describes how multilabel binarizer encoding is utilized and is followed by *subsection D* which provides a description of Gaussian Naive Bayes Classifier and how it is applied to disaggregate energy consumption.

A. Dataset

The dataset used for this experiment consists of the individual energy consumption from many different appliances, the total energy consumption and the date and time measured [21]. For this experiment, we are concerned with the energy consumption of the dishwasher, fridge, microwave, furnace and wine cellar. The state of these appliances is converted to binary. 1 depicting on and 0 for off, which is then further processed to get a label.

The input data considered by the proposed approach is primarily "Total Energy Consumption" from a known set of appliances and time and day of when the consumption is read. From the time and day, we engineered feature such as "Day", "Period of the Day" (morning, afternoon, evening, night) and "Month".

B. Process Overview

The goal of this approach is to gather energy consumption data from different homes with the same set of appliances (preferably having different manufacturers) and generate labels using Multilabel Binarizer to represent appliances combination and their states. We then create a model for each combination of appliances, right after a standard pre-processing of the data. We apply this process to address homes that have subsets of the initial appliances set. This process is shown in high-level in *figure 1*.

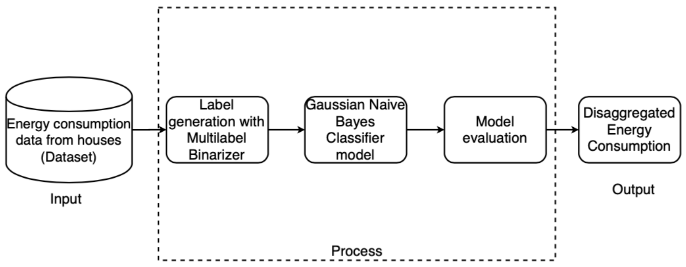

Energy consumption data from houses (Dataset)
Input
Label generation with Multilabel Binarizer
Gaussian Naive Bayes Classifier model
Model evaluation
Process
Disaggregated Energy Consumption
Output

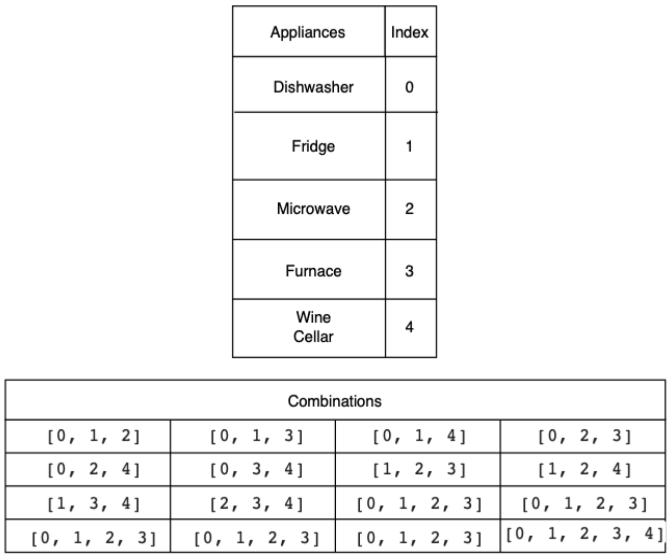
Fig. 1. Process Overview.

The number of possible models we get from this is based on *equation 1* where *n* is the total number of distinct appliances and *r* is for the number of appliances present in the combination. Hence the total number of combinations is controlled by *r*.

$$\text{Total number of models} = \sum_{r=1}^{n} \frac{n!}{r!(n-r)!} \tag{1}$$

Based on the appliances used in our dataset, we chose the minimal number of appliances to disaggregate to be three, and this results in 16 possible model combinations shown in *figure 2* which uses the indices of each appliance.

When the models are obtained, the total energy consumption of each house is directed to the model that was formed



Appliances	Index
Dishwasher	0
Fridge	1
Microwave	2
Furnace	3
Wine Cellar	4

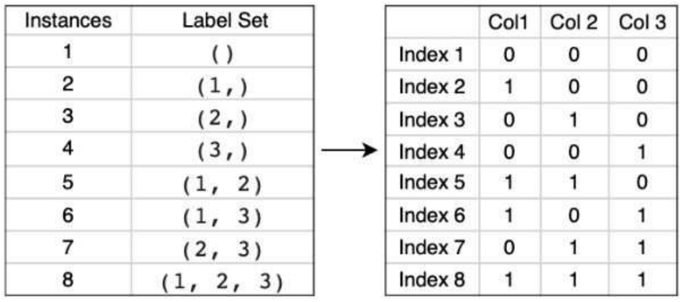
Combinations			
[0, 1, 2]	[0, 1, 3]	[0, 1, 4]	[0, 2, 3]
[0, 2, 4]	[0, 3, 4]	[1, 2, 3]	[1, 2, 4]
[1, 3, 4]	[2, 3, 4]	[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]	[0, 1, 2, 3]	[0, 1, 2, 3, 4]

Fig. 2. Applicance combinations

with the same set of appliances that is seen in the home for disaggregation. With this, the larger the set of the total appliances, the more houses can be covered by allocating the models to homes that have the appliances used to train the model.

C. Multilabel Binarization Encoding

Multilabel Binarizer is a technique used to transform a list of sets or tuples to a matrix of ones and zeros. For a non-empty set or tuple, a 1 is assigned to the specific columns and row, based on the values of the set or tuple and the associated row of the tuple or set. An example mapping is shown in *figure 3*.



Instances	Label Set
1	( )
2	(1,)
3	(2,)
4	(3,)
5	(1, 2)
6	(1, 3)
7	(2, 3)
8	(1, 2, 3)

	Col1	Col 2	Col 3
Index 1	0	0	0
Index 2	1	0	0
Index 3	0	1	0
Index 4	0	0	1
Index 5	1	1	0
Index 6	1	0	1
Index 7	0	1	1
Index 8	1	1	1

Fig. 3. Multilabel Encoding map for three appliances.

This paper utilizes Multilabel Binarization to generate labels for individual and also for combinations of appliances from the universal set of appliances. Each label corresponds to a specific appliance state combination that represents the disaggregation. As an example, in *figure 3*, the number of instances would be the number of models possible given in *equation 1* and for each instance, the appliances would be the columns, and the state combinations would be the indices. For each appliance the possible states could be 1 (On) or 0 (Off). To determine the state, each appliance has a range of energy consumption which indicates whether it is on or off. Moreover, different manufacturers of the same appliance can have different range and, therefore, we have considered a range of threshold for each appliance.

Algorithm 1 creates labels for the data, since there is no direct label assigned to the aggregated readings from the meters. The indices from the matrix of ones and zeros like the example in *figure 3*, is used to represent the states of "on" or "off" in which the numbered index are used as labels. For the number of appliances that form a combination *c* there are 2^c possible different states that could occur. These states are captured in variable map. In lines 10-13 of *Algorithm 1*, for the appliance combination states of on and off, where there is a match in the map that consist of 2^c different states, the index of the map is retrieved. These index values are stored in a vector that results in the same length as the aggregated reading from the meter, when vectorized. As a result we obtain a label vector to create a model with Gaussian Naive Bayes Classifier.

Illustrated in *figure 3*, algorithm 1 describes the use of MultiLabel Binarizer to encode a map for the appliances, while considering all possible combinations.

Algorithm 1. Multilabel Binarizer

```
1)  Input: Binarized state of appliances
2)  Output: A vector of numbers 'n' indicating the label
    combination n € [0,2N - 1] , where N is the number of
    appliances

    Making a Map of state combinations:

3)  For i in range of N:
4)    Make all possible 'i' appliance unique combinations
    from the appliance set
    Considering i appliances:
5)    Make $2^i$ states
6)    If (appliance is on) assign 1
7)    Else assign 0
    The Map consists of 'N' arrays with each array consist-
    ing of all possible state combinations for 'i' appliances,
    where i = 1,2,...,N
    Making an array of indexes from Map for each row
    in data:
8)  If data consist of i appliances, where $i < N$
9)  Go to the 'i appliance array' in map:
10) For j in range length of data:
11)   For n in range length of Map:
12)     If data[j] == Map[n]
13)     return n
```

D. Guassian Naive Bayes Classifier

Gaussian Naive Bayes Classifier (GNBC) is based on Bayes Theorem. In Bayes theorem, we let X be the data tuple which is also known as the evidence. This evidence is described based on a set of n-attributes. Let H be a hypothesis that tuple X belongs to a class C. $P(H|X)$ is the posterior probability and its prior probability is $P(H)$. Similarly, the probability of the evidence, given the hypothesis can be calculated. The posterior probability would then be $P(X|H)$ and its prior probability would be $P(X)$. Bayes' theorem allows to form an equation for calculating the posterior probability $P(H|X)$, from P(H) and $P(X|H)$ from $P(X)$ [22].

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \tag{2}$$

The GNBC works as follows, we let D be a training set of tuples and its classes. Each tuple is formed by 'n'-dimensional feature vector, $X = (x_1, x_2, ..., x_n)$ for features $(A_1, A_2, ..., A_n)$. With 'm' classes, given a tuple X, the classifier classifies the tuple X that belongs to the highest posterior probability that is conditioned on X. From this, the classifier classifies X belonging to a class C_i if and only if [22]

$$P(C_i|X) > P(C_j|X) \quad for \quad 1 \leq j \leq m, j \neq m \tag{3}$$

Maximizing the $P(C_i|X)$ for class C_i is maximizing the posterior hypothesis. By Bayes Theorem, *equation 1* [22]

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)} \tag{4}$$

For Naïve Bayes classification, the naïve assumption of class conditional independence is made. The assumption is that all features are independent of each other, given the label of the class of the feature [22]. Hence,

$$\begin{aligned} P(X|C_i) &= \prod_{k=1}^{n} P(x_k|C_i) \\ &= P(x_1|C_i) \times P(x_2|C_i) \times ... \times P(x_n|C_i) \end{aligned} \tag{5}$$

On top of reducing computation, the probabilities P($x_1|C_i$), P($x_2|C_i$),..., P($x_n|C_i$) can be easily estimated from the training tuples. x_k refers to the value of feature A_k for X. We look to determine whether the feature is continuous or categorical.
1) If it is categorical, then $P(x_k|C_i)$ is the number of tuples with class C_i in all tuples D having the value x_k for A_k, divided by $|C_{iD}|$, the number of tuples of class C_i in D.
2) If dealing with a continuous feature, where A_k is a continuous value, then we need to assume that the continuous value feature follow a Gaussian Distribution with mean μ and standard deviation σ [22].

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp - \frac{(x-\mu)^2}{2\sigma^2} \tag{6}$$

so that

$$P(x_i|C_i) = g(x_k, \mu, \sigma) \tag{7}$$

For this equation, the mean and the standard deviation need to be computed respectively for all values of feature A_k for training tuple of class C_i. The standard deviation and the mean is then plugged in to *equation 6* [22].

We apply the GNBC on our dataset which consists of total energy consumption, timestamp, day, month, period as input features and the numerated indices from the matrix which represent the label, as output. The probability of each class, $P(C_i)$, is computed by taking the number of tuples representing a class, divided by the total number of tuples. With respect to each feature (timestamp and total energy consumption is equivalent to A_k), individual probabilities are computed based on the category of the feature $P(x_k|C_i)$. For this instance, this is applicable to day, month and period since they are represented categorically. For timestamp and total energy consumption, the mean μ and standard deviation σ is computed and plugged into *equation 6*. On a tuple basis, the product of the probabilities is computed and whichever class has the highest probability is the designated class.

To encapsulate the process of how GNBC is used, *Algorithm 2* describes the use of GNBC to fit models and to evaluate the models while also considering all combinations. To recap, line 4 considers only the interested appliances in a given house. Line 7 addresses the change in total energy consumption depending on the combination of appliances. Hence, the total energy consumption feature in the training input is dynamic to the combination. Lines 8-9 in *Algorithm 2* fit the model with the training data and predict with the test data. From the predicted integer values, line 10 gets the states of the appliances, by mapping the predicted integer value to the index of map in *Algorithm 1*. Then the scores are computed.

Algorithm 2. Gaussian Naive Bayes Classifier and model evaluation

```
1) Input: Training Data and Multilabel Binarizer output vector as label
2) Input: Testing Data
3) Output: Model output with number ‘n’ indicating the index of appliance combination in the map where n € [0,2N - 1] , and N is the number of appliances
4) residual = Total energy consumption - Sum of individual appliance energy consumption
5) For i in range of appliance combinations:
6)     Use all possible ’i’ appliance combination to create model
7)     Total energy consumption = Sum of energy consumption of the individual appliances in the combination + residual
8)     Fit model i for i combination using input Training Data
9)     Predict test data with model i for i combination as Ŷ
10)    Get states of appliances from multilabel binarizer map i with Ŷ
11)    For i combination, calculate the score of map values and actual Y
```

IV. Evaluation

This section primarily provides the result of both synthesized and normal test data. 30% of the data is used for the normal test data, the equivalent amount is synthesized for the synthesized test data. We test on the original test data and synthesized data generated from an auto-encoder network to compare the model resiliency. Additionally, using this approach where we have models for different appliance combinations, we do not need to synthesize training data for houses which have a subset of total appliances which is done by Kelly and Knottenbelt [9]. The process used to obtain synthetic testing data is elaborated in *Subsection A*. Followed by *Subsection B* which explains the Jaccard Similarity score for each individual appliance to compare the predicted appliance states (on or off) versus true states. *Subsection C* explains the evaluation metrics used to test the efficiency of the model. Finally *Subsection D* provides the overall scores comparing the predicted and true tuples over the test set by computing accuracy, recall, precision and F1-Score. The result with the synthesized test data in *Table 1* and *Table 3*, can be compared with the results from the non-synthesized test data in *Table 2* and *Table 4*.

A. Synthesized test data with auto-encoder

Auto-encoder is a type of a neural network which tries to reproduce the input data by learning the representation for a set of data in an unsupervised manner [23]. Auto-encoders consist of two components: an encoder to compress the input signal into an encoded representation and a decoder to recreate the input signal. The auto-encoder is designed in a way to restrict it from learning to replicate the input signal exactly, forcing it to prioritize the features to be copied from the input signal in order to create the best approximation for the input. This enables the network to learn useful patterns in the data.

Synthesizing data is fundamentally generating additional data to supplement the original data. There are a variety of ways to synthesize data and a few of them are brought up in [9], [19]. Unless reading meters from multiple homes, we cannot capture the different variance of appliance wattage requirement. Depending on the manufacturer and the year of the make, the wattage consumption of a given appliance may vary as it is proven that appliances are more efficient in recent years [24]. When a new high variance appliance from another manufacturer or different year is introduced, it could result as a residual for the model which does not include that specific appliance. Since our dataset does not include data from multiple homes, data is artificially generated with some noise, with auto-encoder neural network, to test all the model's resiliency. The input is reconstructed using auto-encoders with enough noise. If the newly reconstructed input is the exact same as the input, the experiment becomes purposeless as the initial data could be used instead. In addition, if the newly reconstructed input is too noisy, then important patterns are lost, and it becomes non-comparable as it is obvious that the original model will misclassify. Hence, a balance is needed.

The auto-encoder network is made simple for our utilization purpose. This network consists of an input layer, a hidden layer (encoder) and an output layer. Since we are dealing with continuous data, we segment the data into 24 hours intervals to accommodate the auto-encoder input neurons. The interval between readings from the meter is a minute each, which results to 1440 samples per 24 hours segments to feed to the network for 1440 input neurons. The hidden layer is set to

200 neurons, which is relatively small compared to the input and output layer. With this reduced number of neurons in the hidden layer, the representation of the data are reduced in quality. Hence, noise is introduced when attempting to reconstruct the original data at the output layer with 1440 neurons. This data is then used as testing data and the results are compared with the original test data.

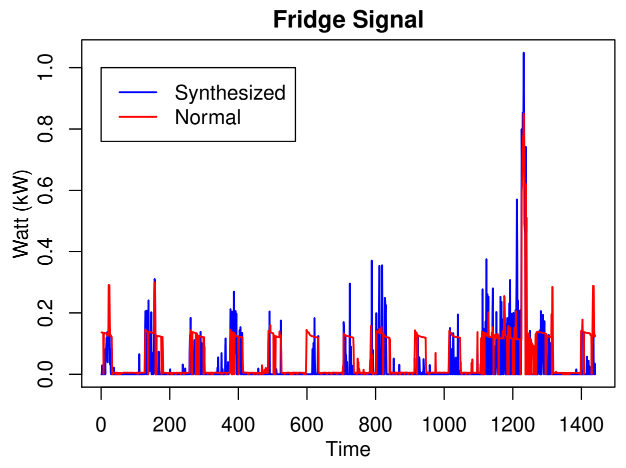


Fig. 4. Synthesized and normal fridge signal comparison

Figure 4 shows the normal signature of the fridge and the synthesized profile that tried to mimic the signature of the fridge. From this pulse-like figure, we randomly replace some of the actual reading with some of the noisy generated reading, where the actual and the generated readings overlap by a certain threshold. By analysis, the thresholds are chosen to exceed a certain wattage to not influence the non-pulsed (the pattern of operation). For the fridge, any overlap of the generated kW and the actual kW that exceeds 0.085 meets the replacement criteria. The same process is done for all the other appliances, for the dishwasher a threshold of 0.6 kW, the furnace and the microwave are set to a threshold of 0.1 kW and the wine cellar's threshold is set to 0.04 kW. Accordingly, when the replacement is done, the aggregated reading is recomputed by including the random replacements and the initial residual mentioned above. From this, we get a day worth of readings as shown in *figure 5*.

B. Jaccard Similarity Score

The jaccard similarity score measures the similarity of two finite sets by overlapping them [25]. The sets do not need to be of the same size. As a result of not being the same size, the similarity measure cannot result to 1. If we let X and Y be the sets, the coefficient can be the intersection of sets X, Y over the union of sets X, Y [25]

$$J(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|}, \qquad (8)$$
$$0 \leq J(X,Y) \leq 1$$

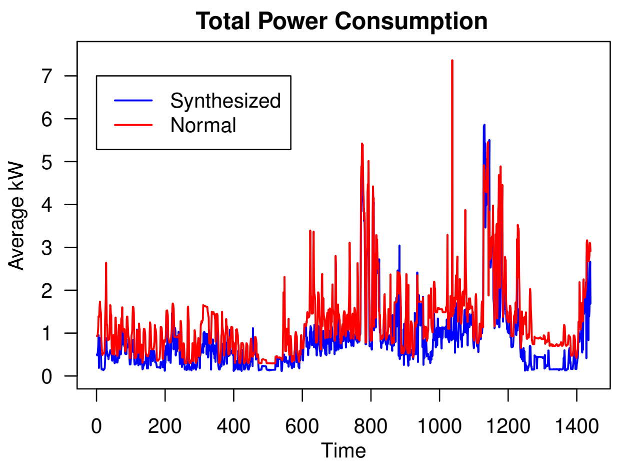


Fig. 5. Synthesized and normal total energy consumption comparison

C. Accuracy, Recall, Precision and F1-Score

Accuracy is a basic metric that returns the number of data points in the test set that are correctly predicted or classified divided by the size of the test set [26]. Precision is a metric to test if the model have the ability to classify a positive sample as positive and not as negative [26]. Recall is a metric to test how well the model is able the capture the actual positive samples [26]. F1-Score is a summarizing metric that allow us to look at the big picture. F1-score is computed by taking the harmonic mean of recall and precision as $2 \times (precision \times recall)/(precision + recall)$ [26].

D. Tabular Results

The models are evaluated in such that results are provided per appliances in tables 1 and 2. Table 1 contains the Jaccard similarity scores obtained from the synthesized readings by the auto-encoder while table 2 consists of the Jaccard scores for non-synthesized data. Each row in both the tables 1 and 2 corresponds to a new model with different set of appliances. For example, to compare the Jaccard similarity scores, the first row in tables 1 and 2 shows the results obtained for model 1 which consists of dishwasher, fridge and microwave. In table 1, model 1 has the Jaccard similarity score of 0.64, 0.81 and 0.82 for dishwasher, fridge and microwave respectively whereas in table 2, model 1 has Jaccard similarity score of 0.65 for dishwasher, 0.89 for fridge and microwave. For the non-synthesized test data, the scores obtained for each appliance is slightly higher which is shown in table 2.

Additionally, an overall score which consider all the appliances of the model is also provided separately in tables 3 and 4. Table 3 contains the accuracy, recall, precision and f1-score obtained from the synthesized readings by the auto-encoder while table 4 contains the same metrics for the non-synthesized data. Each row in both tables 3 and 4 corresponds to a new model with different set of appliances. For example, the first row in tables 3 and 4 shows the accuracy, recall,

TABLE I
JACCARD SIMILARITY SCORE WITH SYNTHESIZED TEST DATA

-	Dishwasher	Fridge	Microwave	Furnace	Wine Cellar
Model 1	0.64	0.81	0.82	-	-
Model 2	0.64	0.81	-	0.82	-
Model 3	0.63	0.81	-	-	0.82
Model 4	0.63		0.82	0.82	-
Model 5	0.65	-	0.82	-	0.82
Model 6	0.64	-	-	0.82	0.82
Model 7	-	0.81	0.81	0.81	-
Model 8	-	0.81	0.82	-	0.82
Model 9	-	0.81	-	0.82	0.82
Model 10	-	-	0.82	0.82	0.82
Model 11	0.63	0.81	0.81	0.81	
Model 12	0.64	0.81	0.82	-	0.82
Model 13	0.64	0.81	-	0.82	0.8
Model 14	0.64	-	0.81	0.82	0.82
Model 15	-	0.81	0.81	0.81	0.82
Model 16	0.63	0.81	0.8	0.81	0.82
Average	0.637	0.81	0.815	0.816	0.818

TABLE III
EVALUATION WITH THE SYNTHESIZED TEST DATA

-	Accuracy	Recall	Precision	F1-Score
Model 1	0.55	0.98	0.80	0.87
Model 2	0.50	0.98	0.79	0.86
Model 3	0.50	0.98	0.80	0.87
Model 4	0.50	0.98	0.80	0.87
Model 5	0.50	0.99	0.80	0.87
Model 6	0.50	0.98	0.80	0.87
Model 7	0.79	1.0	0.87	0.91
Model 8	0.78	1.0	0.87	0.91
Model 9	0.69	0.98	0.86	0.9
Model 10	0.69	0.98	0.86	0.9
Model 11	0.50	0.95	0.85	0.89
Model 12	0.50	0.96	0.86	0.88
Model 13	0.46	0.94	0.83	0.85
Model 14	0.46	0.94	0.83	0.85
Model 15	0.70	0.99	0.87	0.91
Model 16	0.46	0.94	0.84	0.85
Average	0.568	0.973	0.833	0.85

TABLE II
JACCARD SIMILARITY SCORE WITH NON-SYNTHESIZED TEST DATA

-	Dishwasher	Fridge	Microwave	Furnace	Wine Cellar
Model 1	0.65	0.89	0.89	-	-
Model 2	0.65	0.89	-	0.89	-
Model 3	0.65	0.89	-	-	0.88
Model 4	0.65		0.89	0.89	-
Model 5	0.65	-	0.88	-	0.88
Model 6	0.65	-	-	0.88	0.88
Model 7	-	0.89	0.89	0.89	-
Model 8	-	0.89	0.89	-	0.88
Model 9	-	0.89	-	0.89	0.88
Model 10	-	-	0.89	0.88	0.89
Model 11	0.65	0.89	0.89	0.89	
Model 12	0.64	0.89	0.89	-	0.88
Model 13	0.64	0.89	-	0.88	0.88
Model 14	0.65	-	0.89	0.88	0.88
Model 15	-	0.89	0.89	0.88	0.88
Model 16	0.65	0.89	0.89	0.88	0.88
Average	0.648	0.89	0.89	0.885	0.881

TABLE IV
EVALUATION WITH THE NON-SYNTHESIZED TEST DATA

-	Accuracy	Recall	Precision	F1-Score
Model 1	0.58	1.0	0.81	0.89
Model 2	0.52	1.0	0.81	0.89
Model 3	0.52	0.99	0.81	0.89
Model 4	0.52	1.0	0.81	0.89
Model 5	0.52	0.99	0.81	0.89
Model 6	0.52	1.0	0.81	0.89
Model 7	0.8	1.0	0.89	0.94
Model 8	0.8	1.0	0.89	0.94
Model 9	0.72	1.0	0.89	0.94
Model 10	0.72	1.0	0.89	0.95
Model 11	0.52	1.0	0.83	0.91
Model 12	0.52	1.0	0.83	0.91
Model 13	0.47	1.0	0.83	0.9
Model 14	0.47	0.98	0.83	0.9
Model 15	0.72	0.99	0.89	0.94
Model 16	0.47	1.0	0.84	0.91
Average	0.558	0.997	0.842	0.911

precision and f1-score obtained for model 1 which consists of dishwasher, fridge and microwave. In table 3, model 1 has an accuracy of 0.55, recall of 0.98, precision of 0.80 and f1-score of 0.87 whereas in table 4, model 1 has accuracy, recall, precision and f1-score as 0.58, 1.0, 0.81 and 0.89 respectively. As described by many researchers [6], [27], the false negative events usually arise with small appliances. This occurs because appliances with larger energy consumption can overwhelm those with inferior consumption which tends to govern the model. This could lead to larger appliances obtaining more accurate classification while those appliances that consume less perform worse. Although the overall model scores are still provided in tables 3 and 4, it could be misleading to solely rely on the overall scores. For this reason, itemized scores were included in tables 1 and 2.

In the case of multiple homes, for each house the model which has the exact same appliances is selected for disaggregating the total energy consumption of that specific house.

V. CONCLUSION

This paper presents a possible solution to the impact of energy generation on the atmosphere [28] by disaggregating total energy consumption into individual appliances. This can gives consumers information of their appliances usage, for a more efficient use [3], [4]. Energy disaggregation would reveal to consumer which appliances they own are contributing to electricity wastage, whether by having the appliance being on when not in use or using it in the wrong peaks of the day. The disaggregation is obtained through the GNBC models which are created based on the combination of appliances. Prior to fitting the models, multilabel binarizer is used to generate labels. Then for testing, an auto-encoder network is used to produce noisy test data. The models are evaluated with the original and synthesized noisy data, to compare the robustness of the models.

With an average F1-Score of 87.8% on the synthesized test data and an average of 91% on the original data, it shows that it is possible to disaggregate electrical energy in the residential sector relatively well. The scores obtained implies that the models are efficient in disaggregating energy consumption. The performance of the models are vital if we want to use energy disaggregation as a suggestive method to reveal information to consumer. From the scores obtained in the evaluation section, we conclude that the proposed approach performs relatively well and is resilient to some amount of noise. There are still improvements that could be made to our approach.

By analysis, the appliances are not similar in profile which could be the reason the models are resilient on noisy data. Including more appliances will likely worsen the performance of the synthetic results obtained in the result sections. The reason is more appliances makes it more probable for appliance profile overlap, and the addition of noise creates even higher probability of profile overlap. In such case, the same synthesized process could also be applied to the training data, to determine if the model generalize to noisy test data. Additionally, along with some synthesized data, more features can be included as it can enhance the model's performance [29].

References

[1] NRCan, "Energy and greenhouse gas emissions (GHGs)," May 2020. [Online]. Available: https://www.nrcan.gc.ca/science-data/data-analysis/energy-data-analysis/energy-and-greenhouse-gas-emissions-ghgs/20063

[2] G. Akhmat, K. Zaman, T. Shukui, and F. Sajjad, "Does energy consumption contribute to climate change? evidence from major regions of the world," *Renewable and Sustainable Energy Reviews*, vol. 36, pp. 123–134, 2014.

[3] S. Darby *et al.*, "The effectiveness of feedback on energy consumption," *A Review for DEFRA of the Literature on Metering, Billing and direct Displays*, vol. 486, no. 2006, p. 26, 2006.

[4] B. Neenan, J. Robinson, and R. Boisvert, "Residential electricity use feedback: A research synthesis and economic framework," *Electric Power Research Institute*, vol. 3, 2009.

[5] I. Abubakar, S. Khalid, M. Mustafa, H. Shareef, and M. Mustapha, "Application of load monitoring in appliances' energy management–a review," *Renewable and Sustainable Energy Reviews*, vol. 67, pp. 235–245, 2017.

[6] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar, "Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey," Dec 2012. [Online]. Available: https://www.mdpi.com/1424-8220/12/12/16838

[7] K. C. Armel, A. Gupta, G. Shrimali, and A. Albert, "Is disaggregation the holy grail of energy efficiency? the case of electricity," *Energy Policy*, vol. 52, pp. 213–234, 2013.

[8] N. Henao, K. Agbossou, S. Kelouwani, Y. Dubé, and M. Fournier, "Approach in nonintrusive type i load monitoring using subtractive clustering," pp. 812–821, 2015.

[9] J. Kelly and W. Knottenbelt, "Neural nilm: Deep neural networks applied to energy disaggregation," in *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, 2015, pp. 55–64.

[10] K. S. Barsim and B. Yang, "On the feasibility of generic deep disaggregation for single-load extraction," *arXiv preprint arXiv:1802.02139*, 2018.

[11] M. Kaselimi, N. Doulamis, A. Voulodimos, E. Protopapadakis, and A. Doulamis, "Context aware energy disaggregation using adaptive bidirectional lstm models," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3054–3067, 2020.

[12] İ. H. Çavdar and V. Faryad, "New design of a supervised energy disaggregation model based on the deep neural network for a smart grid," p. 1217, 2019.

[13] M. Khodayar, J. Wang, and Z. Wang, "Energy disaggregation via deep temporal dictionary learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 5, pp. 1696–1709, 2020.

[14] P. A. Schirmer and I. Mporas, "Statistical and electrical features evaluation for electrical appliances energy disaggregation," *Sustainability*, vol. 11, no. 11, p. 3222, 2019.

[15] C. Shin, S. Rho, H. Lee, and W. Rhee, "Data requirements for applying machine learning to energy disaggregation," *Energies*, vol. 12, no. 9, p. 1696, 2019.

[16] D. Murray, L. Stankovic, V. Stankovic, S. Lulic, and S. Sladojevic, "Transferability of neural network approaches for low-rate energy disaggregation," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 8330–8334.

[17] H. Liu, Q. Zou, and Z. Zhang, "Energy disaggregation of appliances consumptions using ham approach," *IEEE Access*, vol. 7, pp. 185 977–185 990, 2019.

[18] P. A. Schirmer, I. Mporas, and M. Paraskevas, "Energy disaggregation using elastic matching algorithms," *Entropy*, vol. 22, no. 1, p. 71, 2020.

[19] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 2642–2651. [Online]. Available: http://proceedings.mlr.press/v70/odena17a.html

[20] E. Elhamifar and S. Sastry, "Energy disaggregation via learning powerlets and sparse coding," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[21] T. Singh, "Smart home dataset with weather information," Apr 2019. [Online]. Available: https://www.kaggle.com/taranvee/smart-home-dataset-with-weather-information

[22] J. Han and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2012.

[23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[24] NRCan, "Energy efficiency trends in Canada 1990 to 2013," Jan 2016. [Online]. Available: https://www.nrcan.gc.ca/energy/publications/19030

[25] M. Awad and R. Khanna, *Efficient learning machines: theories, concepts, and applications for engineers and system designers*. Springer Nature, 2015.

[26] A. Sharma, V. R. Shrimali, and M. Beyeler, *Machine Learning for OPENCV4: intelligent algorithms for building image processing apps using OPENCV4, Python, and Scikit-learn, 2nd edition*. Packt Publishing Ltd., 2019.

[27] J. Z. Kolter and M. J. Johnson, "Redd: A public data set for energy disaggregation research," in *Workshop on data mining applications in sustainability (SIGKDD), San Diego, CA*, vol. 25, no. Citeseer, 2011, pp. 59–62.

[28] A. Bridges, F. A. Felder, K. Mckelvey, and I. Niyogi, "Uncertainty in energy planning: Estimating the health impacts of air pollution from fossil fuel electricity generation," *Energy Research & Social Science*, vol. 6, p. 74–77, 2015.

[29] C. Wu and K.-W. Chau, "Data-driven models for monthly streamflow time series prediction," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 8, pp. 1350–1367, 2010.