energies

MDPI

Article

Energy Disaggregation of Type I and II Loads by Means of Birch Clustering and Watchdog Timers

Amitay Kligman, Arbel Yaniv * and Yuval Beck

School of Electrical Engineering, Tel-Aviv University, Tel-Aviv 69978, Israel
* Correspondence: arbelyaniv@mail.tau.ac.il

Abstract: A non-intrusive load monitoring (NILM) process is intended to allow for the separation of individual appliances from an aggregated energy reading in order to estimate the operation of individual loads. In the past, electricity meters specified only active power readings, for billing purposes, thus limiting NILM capabilities. Recent progress in smart metering technology has introduced cost-effective, household-consumer-grade metering products, which can produce multiple features with high accuracy. In this paper, a new method is proposed for applying a BIRCH (balanced iterative reducing and clustering using hierarchies) algorithm as part of a multi-dimensional load disaggregation solution based on the extraction of multiple features from a smart meter. The method uses low-frequency meter reading and constructs a multi-dimensional feature space with adaption to smart meter parameters and is useful for type I as well as type II loads with the addition of timers. This new method is described as energy disaggregation in NILM by means of multi-dimensional BIRCH clustering (DNB). It is simple, fast, uses raw meter sampling, and does not require preliminary training or powerful hardware. The algorithm is tested using a private dataset and a public dataset.

Keywords: balanced iterative reducing and clustering using hierarchies (BIRCH); clustering algorithms; load-disaggregation; non-intrusive load monitoring (NILM); smart grid; smart metering

Citation: Kligman, A.; Yaniv, A.; Beck, Y. Energy Disaggregation of Type I and II Loads by Means of Birch Clustering and Watchdog Timers. *Energies* **2023**, *16*, 3027. https://doi.org/10.3390/en16073027

Academic Editors: Tiago Pinto and Georgios Christoforidis

Received: 26 October 2022
Revised: 8 March 2023
Accepted: 10 March 2023
Published: 26 March 2023

1. Introduction

Smart grid technologies have gained increasing attention in the last ten years [1,2], as electricity demands are rising and they encourage energy supply to be efficient. Earlier research had already indicated that power usage efficiency could grow significantly by providing consumers with frequent feedback about their energy consumption [3]. Other works indicated that smart metering would play a big role in enabling adequate feedback to customers [4,5]. Recent improvements in smart metering technology enable deployment of advanced metering products on a large population scale, hence contributing to the realization of the smart grid vision [6,7] while also encouraging governments to invest a great amount of resources in distributing smart meters over a wide range of households and facilities [8,9]. These widely distributed meters offer new opportunities for implementation of power analysis and feedback techniques.

Non-intrusive load monitoring (NILM) was initially introduced by G. W. Hart [10–12] as a technique intended for identification of load signatures in a single facility or a household via a single measuring point at the entrance of the facility. This technique disaggregates the total power measure into separate components and determines the energy consumption of individual appliances. As modern smart metering technology developed, several approaches were proposed to implement the NILM idea: the factorial hidden Markov model [13,14], particle filter [15], event window [16], deep learning [17], graph signal processing [18], combinatorial optimization [19], modified cross-entropy [20], principle component analysis [21], multi-objective optimization [22], quadratic programming [23], evolutionary algorithms [24], and more. Some of the suggested solutions utilize clustering algorithms suitable for big data analysis [25], especially solutions which can easily be adapted for smart meter reading [26], such as K-means [27], fuzzy

clustering [28], subtractive clustering [29], and more. Some methods utilize an important feature of the V–I trajectory of voltage and current waveforms [30].

Considering functional requirements, an ideal NILM solver should be fast, robust, suitable for big data, and able to handle multiple dimensions (at least three dimensions should be applied). To simplify the discussion, these main criteria were, therefore, chosen for an ideal NILM solver: the linear complexity of $O(n)$, large dataset handling, and the ability to work with low sample rates and process numerical datasets.

Most classic algorithms, such as [19,31], use only real power and cannot support additional features. Multi-feature methods, such as [19,20], show good accuracy and reliability but suffer from high complexity, long computational latency, or an obligated preliminary learning process. Methods that rely on FCM, K-means, or fuzzy clustering [27,28] can achieve linear complexity but only for a fixed number of clusters and limited number of iterations, which means that the number of consumers in the network must be known. Techniques that rely on HMM [14] cannot achieve linear complexity in real time. Newer techniques that rely on deep learning [17,32] or artificial neural networks (ANN) [33] and recurrent neural networks (RNN) [34] suffer from high complexity, high latency, and require significant computing resources for handling big data sets or are limited to work under restricted assumptions, such as CNN with the hypothesis of short time stationarity as a premise of the model [35]. Algorithms that use density-based clustering [36] cannot be implemented in a linear fashion. Algorithms that use graph processing [37,38] cannot achieve linear complexity and usually cannot handle numerical data very well. Another interesting approach utilizes switching transient current measurements [39] but, therefore, requires a high sampling rate for the transient to be measured. A hierarchical clustering-based algorithm was also introduced before [40], but it can only operate at a very high frequency and its ability to handle multiple dimensions is restricted to the harmonics of real and reactive power components.

Based on the selected criteria, and the analysis above, BIRCH would be an appropriate choice for a clustering algorithm, as it is intended for big-data processing [41] and is suitable for multi-dimensional space. It is a dynamic clustering algorithm that incrementally scans the data to form a tree-based structure with no requirements for preliminary knowledge of the data itself, while operating in $O(n)$ complexity and not limited in terms of dimensionality, number of clusters, and numbers of iterations. BIRCH has already been mentioned as a possible clustering algorithm for smart metering applications [42–45] or for energy diagnosis [46]. Former studies, such as [29], imply that BIRCH would be a good clustering choice for load identification, especially when required to handle many samples. However, the method suggested by [43] uses BIRCH as a general technique for learning statistical data from large-scale load profile shapes (not for NILM applications). Although it seems very suitable for unsupervised NILM applications, no previous research has revealed a tested implementation of BIRCH as part of a NILM system in a single facility or a household, using multiple extracted features from a modern smart meter.

This paper introduces a new method for load disaggregation in NILM using multi-dimensional BIRCH clustering (DNB), which is intended for big-data scenarios. This novel idea uses a multi-dimensional raw meter data and implements an algorithm that is specially designed to achieve the requirements of linearity, large-scale data, and numerical nature, in addition to producing good results as a NILM solver. Another important quality is the flexibility of increasing dimensions with raw data from smart meters, enabling more features to be processed (without limiting the feature space to the PQ plane). As is well known, many clustering algorithms, such as k-means [27], spectral clustering [46], ward hierarchical clustering [35], etc., require a known number of clusters. In the case of NILM, the number of clusters is dynamic and therefore these clustering methods cannot be directly implemented and require a long process of optimizing the number of clusters. The core of BIRCH algorithm uses dynamic clustering capabilities to organize multi-featured measurements into clusters within a single scan of the data and therefore a non-pre-determined number of clustering is required. The proposed algorithm is simple, lightweight, can use raw data samples from a smart meter, can use low-frequency sample reading, and does not

require a preliminary training process. The method is applied to type I appliances (on/off devices) and extended by an auxiliary timer to also disaggregate type II appliances (finite state machines). Moreover, the sampled data may contain single or multiple appliances and may also include noisy information. The proposed method was tested using a private dataset, demonstrating a good use of multi-dimensional space. Further tests involved a limited-dimensional scenario, using the AMPds public dataset [47] to compare performance with other NILM algorithms. The results show that DNB performs effectively in comparison with other methods and improves as more dimensions are applied.

The paper is organized as follows: Section 2 formulates the methodology of the NILM process and BIRCH clustering. Section 3 outlines the suggested solution. Section 4 presents simulations and results on a private dataset. Section 5 presents simulation results with comparison to other NILM algorithms. Section 6 concludes the paper.

2. Methodology

2.1. NILM Formulation

Consider a house or facility with m appliances $(i = 1,\ 2,\ \ldots, m)$, each with d features $\psi_{i,1},\ \psi_{i,2}, \ldots, \psi_{i,d}, > 0$. The aggregated features Ψ, measured during sample time n at the entry point of the facility, can be formulated as

$$\Psi[n] = \begin{bmatrix} \Psi_1[n] \\ \Psi_2[n] \\ \vdots \\ \Psi_d[n] \end{bmatrix} = \sum_{i=1}^{m} x_i[n] \begin{bmatrix} \psi_{i,1} \\ \psi_{i,2} \\ \vdots \\ \psi_{i,d} \end{bmatrix} + \begin{bmatrix} \epsilon_1[n] \\ \epsilon_2[n] \\ \vdots \\ \epsilon_d[n] \end{bmatrix} \tag{1}$$

where $x_i[n]$ denotes the *on/off* [0,1] status of appliance i and $\epsilon_1[n], \ldots, \epsilon_d[n]$ represent background noise, such as unmetered appliances or readings that are missing from the disaggregation. It is also assumed that at least one feature is different when comparing two different appliances: $\exists v \vert\ \psi_{i,v} \neq \psi_{b,v}$, for $i \neq b$. The purpose of the NILM algorithm is to disaggregate the measured signal $\Psi[n]$ into its various appliances ψ_i. The resulting optimization problem is formulated as

$$\hat{X}[n] = \underset{x_1[n], \ldots, x_m[n]}{\operatorname{argmin}} \left| \begin{bmatrix} \Psi_1[n] \\ \Psi_2[n] \\ \vdots \\ \Psi_d[n] \end{bmatrix} - \sum_{i=1}^{m} x_i[n] \begin{bmatrix} \psi_{i,1} \\ \psi_{i,2} \\ \vdots \\ \psi_{i,d} \end{bmatrix} \right| \tag{2}$$

where $X[n] = [x_1[n], \ldots, x_m[n]]$ is the solution space. For example, if the features are active power, reactive power, and current phasor angle $(d = 3)$, then (2) may be written as

$$\hat{X}[n] = \underset{x_1[n], \ldots, x_m[n]}{\operatorname{argmin}} \left| \begin{bmatrix} P[n] \\ Q[n] \\ ANG[n] \end{bmatrix} - \sum_{i=1}^{m} x_i[n] \begin{bmatrix} p_i \\ q_i \\ ang_i \end{bmatrix} \right| \tag{3}$$

where $P[n]$ and $Q[n]$ are the aggregated active and reactive power, $ANG[n]$ is the current phasor angle reading at the entry point, and p_i, ang_i, and ang_i are the active power, reactive power, and current phasor angle of appliance i, respectively. In addition, the matrix representing the states of the facility over time is defined as

$$X = \begin{bmatrix} x_1[1] & \cdots & x_1[N] \\ \vdots & \ddots & \vdots \\ x_m[1] & \cdots & x_m[N] \end{bmatrix} \tag{4}$$

Using the matrix representation of (4), and neglecting the noise, (1) can now be visualized as shown in Figure 1.

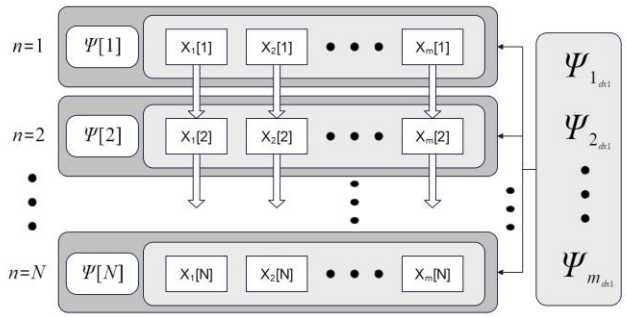

Figure 1. Graphic representation of NILM process.

In a general scenario, multiple appliance states should be considered. Equation (1) can be generalized as a multi-state problem by replacing the single feature $\psi_{i,v}$ with multiple states as options denoted as $\psi_{i,\,j,v}$. Accordingly, the total meter reading at the entry point may be expressed as

$$\Psi[n] = \begin{bmatrix} \Psi_1[n] \\ \Psi_2[n] \\ \vdots \\ \Psi_d[n] \end{bmatrix} = \sum_{i=1}^{m} \sum_{j=1}^{s} x_{i,j}[n] \begin{bmatrix} \psi_{i,j,1} \\ \psi_{i,j,2} \\ \vdots \\ \psi_{i,j,d} \end{bmatrix} + \begin{bmatrix} \epsilon_1[n] \\ \epsilon_2[n] \\ \vdots \\ \epsilon_d[n] \end{bmatrix} \tag{5}$$

where $x_{i,j}[n]$ denotes the status of appliance i. If the appliance operates in mode j, then $x_{i,j}[n] = 1$ and $x_{i,z\neq j}[n] = 0$. Applying this formulation to (2) produces a multi-state optimization problem:

$$\hat{X}[n] = \underset{X[n]}{\operatorname{argmin}} \left| \begin{bmatrix} \Psi_1[n] \\ \Psi_2[n] \\ \vdots \\ \Psi_d[n] \end{bmatrix} - \sum_{i=1}^{m} \sum_{j=1}^{s} x_{i,j}[n] \begin{bmatrix} \psi_{i,j,1} \\ \psi_{i,j,2} \\ \vdots \\ \psi_{i,j,d} \end{bmatrix} \right| \tag{6}$$

where the solution space $X[n] = [x_1[n], \ldots, x_m[n]]$ is a matrix consisting of vectors $x_i \in [0,1]^s$, n denotes the appliance index, n is the sample time, and s is the number of optional modes (states) per appliance.

2.2. Appliance Models

Four appliance models are usually considered [48]:

- Type I—On/off devices: most appliances in households, such as light bulbs and toasters;
- Type II—Finite-state machines (FSM): the appliances in this category present states, typically in a periodical fashion; examples include washers and dryers. Figure 2 displays an example for a type II appliance's active power over time.
- Type III—Continuously varying devices: the power of these appliances varies over time but not in a periodic fashion; examples are dimmers and power tools.
- Type IV—Permanent consumer devices: these are devices with constant power that operate for 24 h, such as alarm systems and external power supplies.

Figure 2 visualizes an example for a type II operation cycle, acquired from a device in the AMPds dataset [47]. This operation can be modeled, in some cases, as a series of on and off events of multiple states over time. This can roughly be considered to be multiple operation events of a type I appliance, as the shift from on state to off state is represented by a drop in most of the active power.

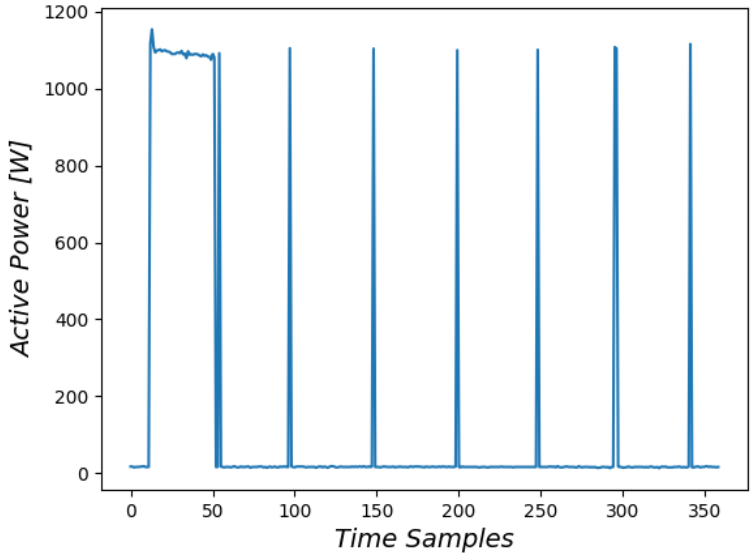

Figure 2. Example of operation cycle of Type II appliance that can be modeled as a series of turn-on and turn-off events. This device's symbol is WOE in the AMPds dataset.

2.3. Reasoning for Adopting BIRCH Algorithm for NILM

As is well known, there are many clustering techniques in the literature [28–30]. Choosing the right algorithm for NILM requires some considerations and the definition of some criteria. Three main criteria should be addressed: (1) linear complexity as a theoretical limit of $O(n)$, (2) enabling analysis of large datasets, and (3) the ability of the method to deal with numerical datasets (and not only categorical or special datasets). Moreover, the algorithm should be easy to implement and not confined to a pre-defined number of clusters. The characteristics of many known clustering methods and their properties are summarized in Table 1.

Table 1. Summary of Clustering Techniques.

Categories	Method	Dataset Size	Type of Dataset	$O(n)$ Complexity?
Partitional	K-Means	Large	Numerical	Yes
	K-Modes	Large	Categorical	Yes
	K-Medoids	Small	Categorical	No
	PAM	Small	Numerical	No
	CLARA	Large	Numerical	No
	CLARANS	Large	Numerical	No
	FCM	Large	Numerical	Yes
Hierarchical	BIRCH	Large	Numerical	Yes
	CURE	Large	Numerical	No
	ROCK	Large	Numerical/Categorical	No
	Chameleon	Large	All types	No
	Echidna	Large	Multivariate	No
Density-Based	DBSCAN	Large	Numerical	No
	OPTICS	Large	Numerical	No
	DBCLASD	Large	Numerical	No
	DENCLUE	Large	Numerical	No

Table 1. *Cont.*

Categories	Method	Dataset Size	Type of Dataset	*O*(*n*) Complexity?
Grid-Based	Wave	Large	Spatial	Yes
	STING	Large	Spatial	No
	CLIQUE	Large	Numerical	No
	OptiGrid	Large	Spatial	No
Model-Based	EM	Large	Spatial	No
	COBWEB	Small	Numerical	No
	CLASSIT	Small	Numerical	No
	SOM	Small	Multivariate	No

The comparison in Table 1 shows that only three clustering techniques are in accordance with the suggested criteria. However, K-means require a predetermined number of clustering and is $O(n)$ for only fixed number of clusters; this is also applicable to FCM. Therefore, BIRCH is shown to have a lot of promise and therefore was chosen to be implemented and tested in this paper.

2.4. BIRCH Clustering Overview

BIRCH was developed in 1996 by a group of researchers at the University of Wisconsin–Madison [37]. It is an unsupervised hierarchal clustering method, which has been developed specifically for data mining and large datasets, especially when the entire data cannot be loaded into memory. It is a dynamic clustering algorithm that incrementally scans data to form a tree-based clustering structure with no requirement for preliminary knowledge of the dataset.

(1) Data Structure

Clustering feature (CF) is a term for quantifying a data-structure that contains a group of data points. It is formally defined as a set of three entities:

$$CF = \left(N, \vec{LS}, SS\right) \tag{7}$$

where N is the number of data points in the structure, $\vec{LS}$ is the linear combination of the datapoints, and SS is the square summation of the data points. $\vec{LS}$ and SS are formally defined as

$$\vec{LS} = \sum_{i=1}^{N} \vec{X}_i \tag{8}$$

and

$$SS = \sum_{i=1}^{N} \left(\vec{X}_i\right)^2 \tag{9}$$

In multi-dimensional cases, (9) should be replaced with a suitable norm.

The above definitions of CF entities allow calculation of quantifying measures essential for describing clustered groups, such as a centroid $\vec{C}$ and a radius R:

$$\vec{C} = \frac{\sum_{i=1}^{N} \vec{X}_i}{N} = \frac{\vec{LS}}{N} \tag{10}$$

$$R = \sqrt{\frac{SS}{N} - \left(\frac{\vec{LS}}{N}\right)^2} \tag{11}$$

A CF-Tree is a height-balanced tree data structure which is built while serially scanning the data. It can be described as a tree-like representation of the data points. The structure is based on two parameters: branching factor (B), which represents the number of children for each non-leaf node, and threshold (T), which represents the radius of maximal distance for data points in a single cluster. The threshold parameter actually determines the tree size. A CF-Tree has two types of nodes: the first type is a non-leaf node, which contains at most B entries of the form $[CF_i, child_i]$, where $child_i$ is a pointer to the ith child node and CF_i is the clustering feature, representing the associated sub-cluster. The second type of node is a leaf node, which contains at most L entries—each of the form $[CF_i]$. It also has two pointers: prev and next, which are used to chain all the leaf nodes together. Figure 3 visualizes the CF-Tree structure. The CF-Tree data structure is therefore a very compact representation of the dataset because each entry in a leaf node is not a single data point but a sub-cluster.

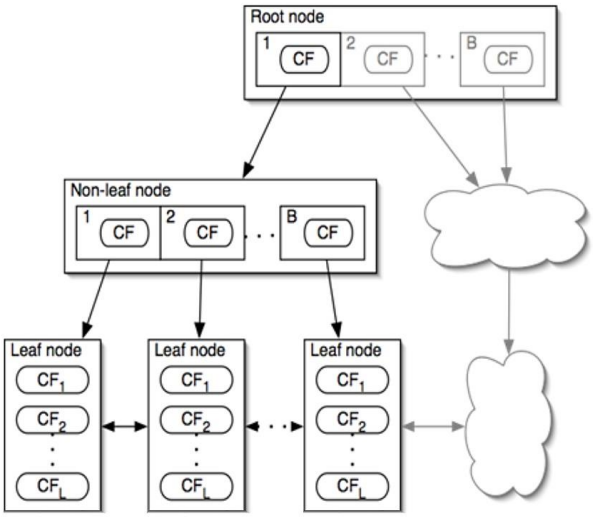

Figure 3. A graphical representation of the CF-Tree. Each cluster feature (CF) record represents a cluster or a sub-cluster of datapoints.

(2) BIRCH Algorithm

BIRCH is described as a four-step algorithm. The first step is an insertion algorithm, intended for creation and update of the CF-Tree, based on a serial scan of the data. This is the first and most critical step of the BIRCH algorithm. Each newly encountered data point causes the CF-Tree to be traversed, starting from the root and choosing the closest node at each level (calculated from the CF record of the node). If the closest leaf cluster for the new data point is finally identified, a test is performed to check two criteria: first, the leaf must still satisfy the threshold (T), as required from the radius definition in (11); second, the size characterization of the leaf (L) must not be exceeded. Two options are therefore possible: if the sample can be inserted within the threshold and size requirements then CF records would be updated accordingly. However, if the sample cannot be inserted due to the size requirement, the leaf would then be split, and its samples are therefore reassigned (this may result in further splitting at the parent level based on the number of entries in the non-leaf nodes). In each of the two described outcomes, the new entry will cause the tree to update all the CF records for the changed leaf nodes and non-leaf nodes traversed to the root.

The second step (optional) is condensing the CF-Tree into a smaller form by increasing T. The third step is to apply clustering on the leaf nodes, which can be scanned without traversing the tree, based on their next and prev pointers. The fourth step (optional) will be further refining by reassigning data points to the closest centroids from the third step, based on the centroid definition in (10).

An important advantage of the BIRCH algorithm is that it only requires a single scan of the database. Compared to the popular K-means algorithm, BIRCH performs exceptionally

well in terms of less memory consumption, faster performance, less order-sensitivity, and high accuracy [49]. A main disadvantage of BIRCH (and of most hierarchal clustering methods) is less robustness towards outliers, which will either show up as additional clusters or even cause other clusters to merge.

3. BIRCH for Power Applications

3.1. Solution Overview

(1) Requirements

Figure 4 simplifies the requirements of the algorithm. The process acquires its data as features-over-time, extracted from raw low-frequency meter samples. Each feature received by $\Psi_i[n]$ is as denoted in (1) for each sample n. The algorithm output should be a sorted list of devices vectors, such that each vector is a record containing pairs of "event time" and "toggle status" (turn-on or turn-off) for all toggle events in the data. The process also requires the output data to be suitable for (optional) post-processing so that better results can be obtained. Extraction of evaluation metrics, such as F-measure [50] and TPCA (total power currently assigned) [51], can be performed on the output in order to have some kind of assessment about the performance of the algorithm.

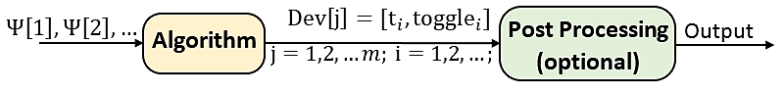

Figure 4. High-Level Process Overview. Post-processing is essential for measures and performance evaluation.

The algorithm output *Dev[j]*, as denoted in Figure 4, is a simplified way of describing the solution $X[n]$ for an on–off solution space, as denoted in (2), and is also referred to as a type I load. The optimal solution would be denoted as $\hat{X}[n]$. In (2), the entity $x_i[n]$ represents an indication for whether the ith device is in the on or off state at the time event of sample n. For simplicity, the off state can be regarded as the state where most of the appliance's on-power is not indicated. A type II appliance could therefore shift back and forth from on state to off state during its activation cycle, as was shown above in Figure 2. The type of appliance is considered as well, by adding an auxiliary timer, as will be shown below.

(2) Input data

The input is a series represented by sampled features of d dimensions. Namely, each sample is considered a d size vector, as formulated in (1). An off-event is defined as a drop in the majority of the consumer's active power; therefore, it requires at least one of the features to be an active power measurement or a feature that follows its behavior. It is, therefore, assumed that one of the $\Psi_i[n]$ features is an active power reading or linearly proportional to it; hence, $\exists i,\ \Psi_i[n] \propto P[n]$. For convenience, it will be assumed that $\Psi_1[n] = P[n]$ so that $\Psi_1[n] = P[n]$. Thus, according to (1), the input vector can be written as

$$\Psi[n] = \begin{bmatrix} \Psi_1[n] \\ \Psi_2[n] \\ \vdots \\ \Psi_d[n] \end{bmatrix} = \begin{bmatrix} P[n] \\ \Psi_2[n] \\ \vdots \\ \Psi_d[n] \end{bmatrix} \tag{12}$$

and the active power is, therefore, known for each sample n.

3.2. DNB Process Phases

The process is iterative and comprises four essential steps as depicted in Figure 5. Each iteration handles a single toggle event. There are three predefined configuration parameters:

- Dimensions selection instructs which features to choose from the raw data (depending on the meter's model capabilities);

- Weight parameters instructs how to determine (or to define) an event occurrence during the time samples' scan;
- Clustering parameters, such as the threshold T, which have been elaborated in an earlier section.

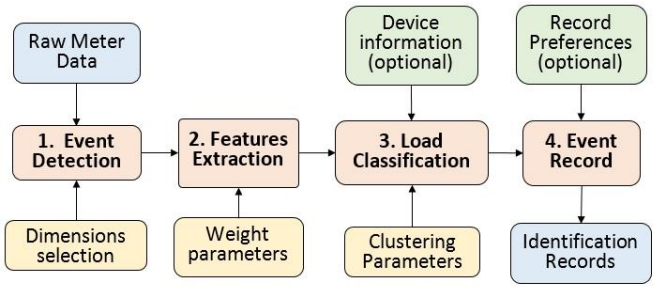

Figure 5. DNB Block Scheme. The process comprises four main steps.

These three selections, presented as yellow blocks in Figure 5, could be further optimized (optional) for the data input scenario, in terms of meter sample rate, meter available features, dataset scale, appliance characteristics, etc.

There is also room for optional input from the user, which can add information for the classification and to the record, such as user-defined labeling.

Raw input data $\Psi[n]$, from the meter, can also be described as a stream of samples so that some of them represent events and will be denoted by t_i (for the ith event). These events are separated by non-event time samples, denoted by τ_0:

$$input = (t_1, \tau_0, \ldots, \tau_0,\ t_2, \tau_0, \ldots, \tau_0,\ t_i, \tau_0, \ldots) \tag{13}$$

The number of τ_0 separating each of the t_i occurrences can be equal or greater than zero. Thus, the goal of the four-stage process is to identify, classify, and record these time samples n, which represent an event t_i; hence $n_i = t_i$, where n_i is the exact time sample of the occurrence of event t_i. Therefore, each single iteration i handles a single time event t_i.

An optional later step relates to the post-processing, mentioned earlier in Figure 4. It requires additional information, such as power consumption estimation, or some specific tuning of the algorithm parameters.

Next, the four steps of the algorithm are described.

(1) First Step: Data Acquisition and Event Detection

Given raw data as a series of d-dimensional features $\Psi[n]$, as denoted in (1), it is assumed that a change in the features' value is present at the event time, so changes in $\Psi[n]$ are inspected as:

$$\Delta\Psi[n] = \begin{bmatrix} \Psi_1[n] \\ \Psi_2[n] \\ \vdots \\ \Psi_d[n] \end{bmatrix} - \begin{bmatrix} \Psi_1[n-1] \\ \Psi_2[n-1] \\ \vdots \\ \Psi_d[n-1] \end{bmatrix} = \begin{bmatrix} \Delta\Psi_1[n] \\ \Delta\Psi_2[n] \\ \vdots \\ \Delta\Psi_d[n] \end{bmatrix} \tag{14}$$

For the event detection, a detection function that performs a weighted calculation of the changes in features is considered:

$$f(n) \equiv f(\Delta\Psi_1[n], \Delta\Psi_2[n], \ldots, \Delta\Psi_d[n]) \tag{15}$$

The output of the weighted calculation $f(n)$ is Boolean in order to differ the *detected* case from the not-detected case:

$$f(n) = \begin{cases} true & event\ detected \\ flase & event\ not\ detected \end{cases} \tag{16}$$

The required outcome of step 1 is to produce an event time t_i, which represents the ith event in the dataset series, hence:

$$if\ (f(n) = true):\ t_i = n \tag{17}$$

(2) Second Step: Feature Extraction

The purpose of this step is to create a feature-description of the detected event, using a time-vector, denoted as $Stream_i$, which contains data to represent the specific event that was detected in the last step. For the t_i retrieved in step 1, a time interval is defined as:

$$\Delta T_i = min\begin{cases} t_i - t_{i-1} \\ t_{i+1} - t_i \\ \Delta t_{default} \end{cases} \tag{18}$$

where $\Delta t_{default}$ is a constant value intended to limit ΔT_i according to the test scenario and meter sampling rate.

The vector $Stream_i$ is constructed from ΔT_i components and defined as

$$Stream_i = [Stream_i[0],\ Stream_i[1], \ldots, Stream_i[\Delta T_i]]$$

where each component $Stream_i[k]$ is defined for an integer value of $k = 0, 1, 2, \ldots, \Delta T_i$, such that:

$$Stream_i[k] = \begin{cases} \Psi_1[t_i + k] - \Psi_1[t_i - \Delta T_i + k] \\ \Psi_2[t_i + k] - \Psi_2[t_i - \Delta T_i + k] \\ \vdots \\ \Psi_d[t_i + k] - \Psi_d[t_i - \Delta T_i + k] \end{cases} \tag{19}$$

$Stream_i$ can be described as a series of samples, acquired from a time interval of $2{\cdot}\Delta T_i$, as depicted in Figure 6. This list can be regarded as a local feature-description of the event t_i.

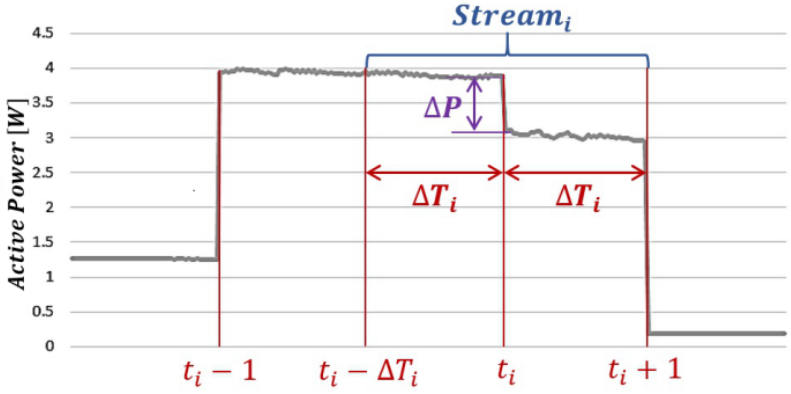

Figure 6. Example of data-stream extraction for a single feature. The stream is acquired from an interval with length of 2·ΔT_i.

The method in Figure 6 represents the extraction of $Stream_i$ for a single dimension (as an example). However, in the common scenario, multiple dimensions are involved. Thus, the outcome of step 2 is a stream of extracted features that represents the event t_i in the features' dimensional space.

(3) Third Step: Load Classification

$Stream_i$, that was extracted in the last step, can now be appended to other streams from the last iterations to form a combined stream of data for clustering:

$$Stream = [Stream_0,\ Stream_1, \ldots, Stream_i] \tag{20}$$

BIRCH clustering is applied to $[Stream]$ to produce an updated CF-Tree. With an updated CF-Tree, the current event, detected in step 1, can now be interpreted in terms of a cluster ascription by traversing the CF-Tree from the root to the most suitable leaf. Such a suitable leaf must be available since the clustering must have formed an appropriate cluster (if it did not already exist). There are now two options:

- If a new device is detected, a new device affiliation is created (as a new cluster) and the event at t_i is identified as part of the new affiliation.
- If an existing device is detected (namely, there is already an appropriate cluster) the event at t_i is identified as part of an existing cluster.

User information can also be considered (optional), as depicted in Figure 5. For example, the user can decide that a certain event is not required to be clustered in order to discard it from the records (in order to improve performance, ignore outliers, and reduce error).

The outcome of the third step is affiliation of the event at t_i to a cluster, hence to an appliance state, denoted in (5) as $\psi_{i,j,x}$.

(4) Fourth Step: Event Data Record

The appliance state information, retrieved in step 3, must be documented and appended to the device record in order to form a complete record of the events' operation. For each device, a list of the events' times and toggles is created: the device record to-be-updated is known from step 3 as the identified cluster. The event time is known from step 1 as the n which caused the outcome $f(n) =$ true and initiated feature extraction in step 2 for the event time $t_i = n$. The event toggle will be set as either a *turn-on* or *a turn-off*, based on the following criteria: if $\Delta P[n] > 0$ then the toggle event is "*turn-on*", denoted as $toggle_i = 1$, and when $\Delta P[n] < 0$ then the toggle event is "*turn-off*", denoted as $toggle_i = -1$.

The process output, therefore, comprises lists of devices: *Dev [1]*, *Dev [2]*, . . . , *Dev[m]*, each containing a detailed record of pairs: $[toggle_i,\ t_i]$. This form is a simplified denotation for the NILM problem solution explained earlier.

3.3. DNB Post-Processing (Optional)

The output analysis is a separate optional step, which has a significant role in analyzing and presenting the algorithm output. In this stage, all supplementary information are added, such as estimated power-per-appliance values.

(*1*) Considering Appliance Type

The appliance type should also be considered as part of the post-processing. As described earlier, DNB's identification abilities are based on detection and clustering of events. In terms of load type, the algorithm focuses on type I appliances. Namely, the detection of type II appliances can be considered as a series of turn-on and turn-off events (based on power changes). If detection of type IV is required, it must rely on preliminary knowledge of the start state, since the algorithm only finds feature changes and type IV appliances usually do not have a distinct turn-on event throughout the dataset.

(*2*) *Auxiliary Timer*

Event detection, based on the weighted parameters function in (15), is the Achilles' heel of recordkeeping in step 4, since the detection of a turn-on event followed by a miss of a turn-off event would cause the appliance to be "stuck" as on, producing a significant difference between estimated and real energy consumption. This problem is adequately resolved by an auxiliary timer, operating in a watchdog timer (WDT) fashion. It is assumed that any normally off device operation cycle (especially type I and II) includes a series of at least one pair of turn-on and turn-off events; therefore, given a determined amount of time, at least one toggle event is expected. If such an event was not detected, the WDT would

expire and trigger a device "reset" by adding a device turn-off event at the expiration time. Figure 7 describes the implementation of the auxiliary timer.

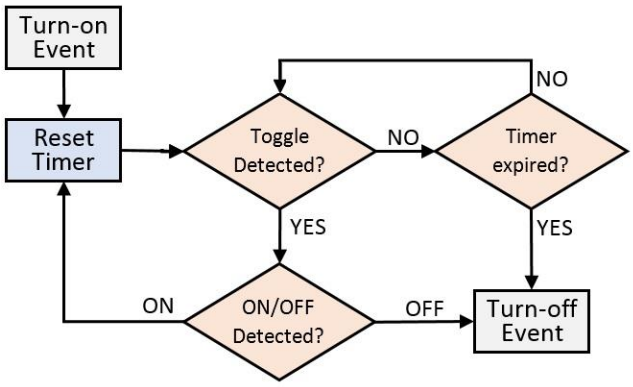

Figure 7. Auxiliary timer implemented as a watchdog timer. The timer loop is initiated by a turn-on event and terminated by a turn-off event.

WDT expiration time value may be either global for all appliances or individual per appliance.

3.4. Multi-Dimension and Feature Selection

As mentioned earlier, DNB is very suitable for multi-dimensional clustering. Adding dimensions has clear benefits in terms of identification: first, there are more features to be weighted for (15); second, there are more components of $\Delta\Psi$ to be clustered in the data stream, as deduced from (19).

However, adding more and more dimensions might be redundant and might also cause problems such as the curse of dimensionality [52] and overfitting [53]. Therefore, selection of the required features must be toward the minimal number of features that hold maximal useful information. As a rule of thumb, features that show a distinctive difference from one appliance to the other are preferred.

As an example, Table 2 specifies the features of appliances that were measured in the lab. Disaggregation of the separated appliances will not be possible with the classic two-dimensional selection of active and reactive power (*P*–*Q* plane), since Light 1 and Light 2 are very similar in active and reactive features. The data in Table 2 were collected in the lab using SATEC PM135EH smart meter.

Table 2. Example: Features Estimation for Lights Appliances.

Appliance	Avg. *P* [kW]	Avg. *Q* [kVAR]	Avg. *ANG* [Deg]
Light 1	0.55	0.43	−15.54
Light 2	0.89	0.61	−34.05

An estimation of Euclidian distance between the two-dimensional data points would be:

$$\sqrt{(P_{L1} - P_{L2})^2 + (Q_{L1} - Q_{L2})^2} = 0.38$$

Adjusting the scale, by multiplication of P and Q with scalars, would cause other identification problems due to the noisy envelope of Light 1 (noticeable in Figure 8). This matter is solved by adding the third feature of the current phasor angle, which can be extracted from the raw meter reading by simple phasor math and is stable enough to allow adjustment of the scale without noticeable noise. An estimation of Euclidian distance between the three-dimensional data points would be:

$$\sqrt{(P_{L1} - P_{L2})^2 + (Q_{L1} - Q_{L2})^2 + (ANG_{L1} - ANG_{L2})^2} = 1.7421$$

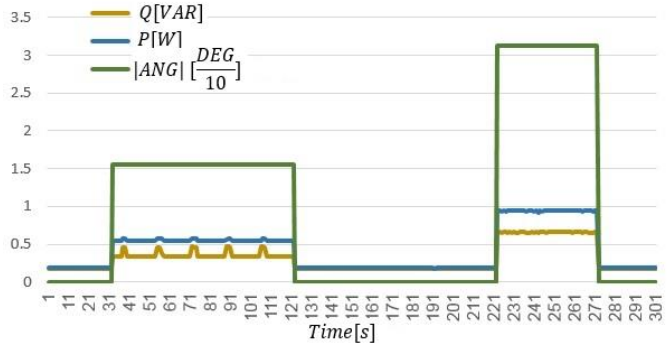

Figure 8. Feature graph of Light 1 and Light 2, which are difficult to disaggregate with two features but easily disaggregated with three. *P* is active power, *Q* is reactive power, and *ABS(ANG)* is absolute phasor current angle.

The matter is further illustrated in Figure 8, as the current phasor angle was divided by 10 in order to adjust to the graph scale. In this specific case, one feature *ANG* would probably be enough for disaggregation. However resistive appliances (kettle, toaster, etc.) are usually characterized by mainly active power and a very small angle for the fundamental component [42]. Therefore, in some cases, choosing a current phasor angle alone would not be sufficient. Moreover, as explained above, the algorithm requires at least one feature that is proportionally following the active power *P* to be selected. Although information of the toggle type is easily extracted from active power differences, it cannot confidently be extracted from the current phasor angle.

4. Private Dataset Simulations

An experiment on a private dataset is included in this study in addition to a public dataset. The use of private datasets is not common in NILM research as it prevents reproduction of the results. However, the necessity of a private dataset in this study is to demonstrate a raw multiple-features input using a modern smart meter, in addition to being a good example of the use of feature selection.

(1) Settings and test scenarios

This experiment was conducted in a lab, covering 1800 m samples from various appliances, which were measured separately, as well as combined with each other in multiple combinations, before being joined together into a single data series. No prior training process was executed. The private dataset was collected in a typical operation manner using the SATEC PM135EH smart meter. The data were acquired at a low sample-rate of 1 Hz. The selected smart meter measures active power, reactive power, apparent power, current, power factor, THD, current phasors, and harmonics. All these features are suitable, of course, for DNB application.

In this experiment, six appliances were used. The environment was not fully de-noised, as some low-power appliances were still connected during measurements, indicated as noticeable measurement-noise. The power features used were active power (P), reactive power (Q), phasor current amplitude (I), phasor current angle (ANG), and total harmonic distortion (THD), as presented in Table 3.

Preparations for this test included dimensions selection as active power (*P*), reactive power (*Q*), and current phasor angle (*ANG*). DNB parameters were then optimized to produce maximum identification of measured appliances.

Table 3. Features Estimation for Devices (Private Dataset).

Appliance	Avg. P [kW]	Avg. Q [kVAR]	Avg. THD [%]	Avg. I [A]	Avg. ANG [Deg]
Microwave	1.061	0.130	34.632	4.806	−2.303
AC	2.947	1.563	17.882	4.884	−25.466
Kettle	1.869	0.002	2.28	8.151	−1.383
Light 1	0.366	0.253	26.449	1.312	−15.544
Light 2	0.706	0.433	20.632	3.924	−34.046
Computer	0.155	0.017	25.438	0.785	−7.851

(2) Results Analysis

The feature space of the selected dimensions is presented in Figure 9 as a three-axis graph, each axis represents one $\Delta\Psi_i[n]$ component as formulated in (14). The datapoints in the graph represent the acquired samples stream, denoted as the list $[Stream]$ (20). Two of the features are calculated from active and reactive power. The third feature quantifies the absolute value of the appliance phasor angle, adjusted to the graph scale as

$$ACC\_IANG = |\theta \cdot 0.12| \tag{21}$$

where θ is the phasor angle, given in degrees; the value 0.12 was chosen and optimized by trial and error. The difference in phasor angle is computed for the event moment by simple phasor math using the phasor current amplitude and phasor current angle, which was acquired by the meter.

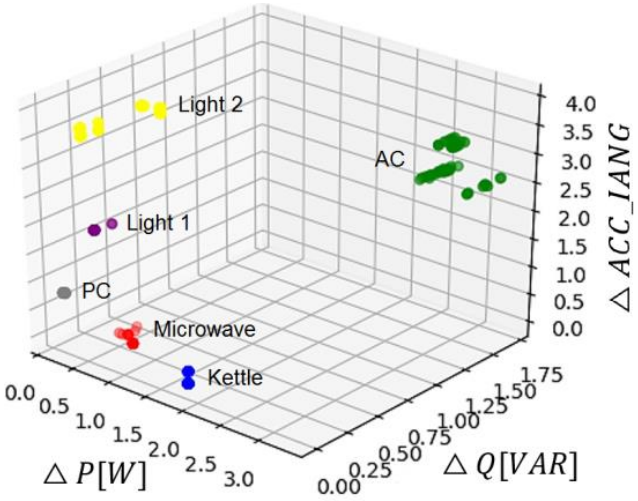

Figure 9. Three-dimensional results: AC (green), Light 1 (purple), Light 2 (yellow), Kettle (blue), Microwave (red), Computer (gray). Delta P is active power difference, Delta Q is reactive power difference, Delta ACC_ANG is the scaled phasor angle difference.

Figure 10 is a simple orthogonal projection of the datapoints in Figure 9 onto a P–Q plane. This two-dimensional display keeps the colors of the original clusters as presented in Figure 9.

It is clear from the arrangement of the datapoints in Figure 10 that, without the third dimension, Light 1 and Light 2 would be mixed into the same cluster, since the two features merge in the *P*–*Q* plane. The microwave and Light 2 would have their centroids adjacent, which would, allegedly, be solved by different clustering parameters but in practice could cause additional problems, such as dividing other clusters due to the threshold radius being too small. This problem also applies to the Light 2 appliance with the computer appliance. The AC appliance is clearly separated by both *P* and *Q* features.

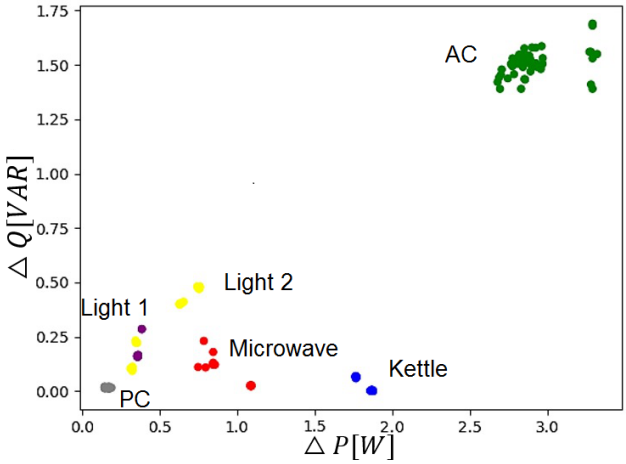

Figure 10. Two-dimensional projection: AC (green), Light 1 (purple), Light 2 (yellow), Kettle (blue), Microwave (red), Computer (gray). Delta P is active power difference, Delta Q is reactive power difference.

5. Public Dataset Simulations

5.1. Evaluation Metrices

In NILM algorithms, there are several options for evaluations metrics intended for determining a classification score. Most of these metrics compare NILM algorithms as binary classification tasks, with four possible outcomes: true positive (TP)—the number of times an appliance is correctly detected as on; true negative (TN)—the number of times an appliance is correctly detected as off; false positive (FP)—the number of times an appliance is wrongly detected as on; and false negative (FN)—the number of times an appliance is wrongly detected as off. Accuracy is a commonly used evaluation metric and is defined as

$$Accuracy = \frac{TP + TN}{TP + TN + TP + FN} \tag{22}$$

The accuracy metric is inefficient in the case of NILM problems due to the accuracy paradox [54]. Therefore, a popular metric from the information retrieval domain, named F-measure (also known as F1-score) [50], is utilized, which expresses the harmonic mean of *precision* and *recall*, given by

$$F - measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{23}$$

where

$$Precision = \frac{TP}{TP + FP}, \; Recall = \frac{TP}{TP + FN} \tag{24}$$

Precision is defined as the correct classification of all positive estimations. Recall is defined as the percentage of on-state appliances that are correctly recognized.

In addition, another important metric is the total power correctly assigned (TPCA) [38], which is defined as how much of the power consumed by an appliance can be assigned correctly during N samples. TPCA is calculated as

$$TPCA = 1 - \frac{\sum_{n=1}^{N} \sum_{i=1}^{m} |p_i[n] - \hat{p}_i[n]|}{2 \cdot \sum_{n=1}^{N} P[n]} \tag{25}$$

where $p_i[n]$ is the actual power consumption of the ith appliance at sample time n, $\hat{p}_i[n]$ is the estimated power consumption of the ith appliance at sample time n, and $P[n]$ is the aggregated power measured during sample time n. Thus, the TPCA metric quantifies the percentage for correct power draw estimation.

5.2. Settings and Test Scenarios

(1) Public Dataset

There are several popular public datasets available for NILM studies. Some popular datasets, such as REDD [51,55], only use active power measurement and, therefore, are not suitable for DNB application. Another popular dataset is [56], that contains only several days of household current and voltage readings and does not hold enough data for a proper low sampling-rate experiment.

AMPds is an open dataset proposed by SFU [47] The data samples were collected at a house in Canada over 1 year. AMPds contains low sample-rate recordings of 60 s per measurement, for 21 sub-meters, and includes active, reactive, and apparent power, in addition to current and voltage readings. AMPds is an exception in terms of datasets as different features are also provided for appliance measurements. In this simulation, six sub-meters were classified—each contained a single appliance. The clustered meter features were active power and reactive power. The scenario contained 600 h of noisy measurements (25 days). No preliminary training process was applied. The start time of the scenario was 5 April 2012, 15:38:00. That specific time instance and duration was chosen to comply with the exact scenario of compared studies [20–22].

As explained in [34], AMPds meter samples are based on two sensor readings (voltage and current), which enable calculation of all other features. Thus, only two dimensions are useful, whereas any other added feature would not provide useful information. Hence, the AMPds dataset actually limits the abilities of DNB, since it can provide only two features to construct a two-dimensional space for clustering and, consequently, limits the number of loads to be identified. This disadvantage was considered when choosing AMPds. Nevertheless, it provides a good base for comparison with the results of other works.

(2) Compared Methods

DNB was compared to other methods that were also intended to solve NILM: combinatorial optimization (CO) [20], modified cross-entropy (MCE) [21], and principle component analysis (PCA) [22]. CO finds the combination of appliance states, which minimizes the difference between the sampled aggregated features and the summation of known appliances features, subject to a set of states (on/off), as shown in (6). MCE classifies NILM events based on the cross-entropy method. PCA is a technique for dimension reduction that projects features on lower dimensional space in order to optimize other methods (as CO and MCE). In [22], MCE and CO are compared with and without the use of PCA. In this study, the dataset range was chosen according to [21] for comparison. In AMPds simulations, the average features are those shown in Table 4.

Table 4. Power Estimation for Devices (AMPds Dataset).

Appliance	Avg. P [W]	Avg. Q [VAR]
Dishwasher (DWE)	770	40
Dryer (CDE)	4613	425
Fan thermostat (FRE)	370, 110	26
Heat Pump (HPE)	1810	358
Kitchen Fridge (FGE)	127	0
Oven (WOE)	3570	120

5.3. Experimental Results

The results of applying AMPds scenarios to DNB are shown in Table 5 and Figures 11 and 12. The graph in Figure 11 contains the results of 32 h (starting at the 65th h of AMPds), whereas the results of Table 5 and Figure 12 are for the entire scenario of 600 h, starting from the 110th h. The bar chart in Figure 12 shows the accuracy performance of assigning estimated energy-per-appliance to the algorithm output versus the ground truth energy consumption. As can be seen, assignment of estimated energy is in proximity

to the ground truth, and most of the difference can be found in the HPE (69.88% instead of 53.31%). Much of the difference is attributed to the mis-identification of FGE, which is the most difficult appliance to be identified, since its initial power difference is not consistent for multiple measurements. It can be seen from the results in Table 5 (for both accuracy and F-measure) that the DNB technique is robust and shows good results at a low sampling rate in contrast to handling a large dataset. A comparison to other works is presented in Tables 6 and 7. The results in Table 6 are compared to the MCE and CO methods, with either one or two features. The results in Table 7 are compared to the PCA method, utilizing MCE and CO with either one or two features. Since the TPCA calculation in [21] was for seven appliances, the TPCA calculation of DNB was modified by adding in a zero-power reading for the seventh appliance, listed as TPCA (2) in Tables 6 and 7. In other words, DNB was tested under tighter conditions, as TPCA (2) was formally calculated from (25), by adding the true total power of the seventh appliance to $\sum_{n=1}^{N} P[n]$ in the formula's denominator and adding $\hat{p}_i[n] = 0$ in the numerator, as the estimated power consumption of the seventh appliance for all time samples.

Table 5. Simulation Results Summary.

Appliance	Accuracy	Precision/Recall	F-Measure
WOE	0.9987	0.833/0.856	0.8446
HPE	0.9801	0.958/0.891	0.9232
CDE	0.9944	0.690/0.782	0.7331
DWE	0.9834	0.638/0.533	0.5806
FGE	0.9919	0.315/0.824	0.4561
FRE	0.9985	0.998/0.999	0.9992
AVG	0.9912	0.739/0.814	0.7561
TPCA	0.8313		

Table 6. Comparison to MCE and CO Methods.

Appliance/Features	DNB *P, Q*	MCE *PC1*	MCE *PC1, PC2*	CO *PC1*	CO *PC1, PC2*
WOE	0.845	0.247	0.390	0.126	0.342
HPE	0.923	0.255	0.226	0.107	0.261
CDE	0.733	0.406	0.564	0.300	0.562
DWE	0.581	0.445	0.517	0.163	0.450
FGE	0.456	0.602	0.742	0.140	0.074
FRE	0.999	0.619	0.912	0.259	0.128
AVG F1	0.756	0.429	0.559	0.183	0.303
TPCA (2)	0.801	0.715	0.788	0.361	0.583

Table 7. Comparison to MCE and CO Methods with PCA Technique.

Appliance/Features	DNB *P, Q*	MCE *PC1*	MCE *PC1, PC2*	CO *PC1*	CO *PC1, PC2*
WOE	0.845	0.510	0.654	0.184	0.340
HPE	0.923	0.264	0.206	0.235	0.229
CDE	0.733	0.852	0.923	0.704	0.541
DWE	0.581	0.051	0.500	0.420	0.477
FGE	0.456	0.768	0.313	0.386	0.403
FRE	0.999	0.803	0.946	0.705	0.786
AVG F1	0.756	0.541	0.590	0.439	0.463
TPCA (2)	0.801	0.719	0.760	0.616	0.701

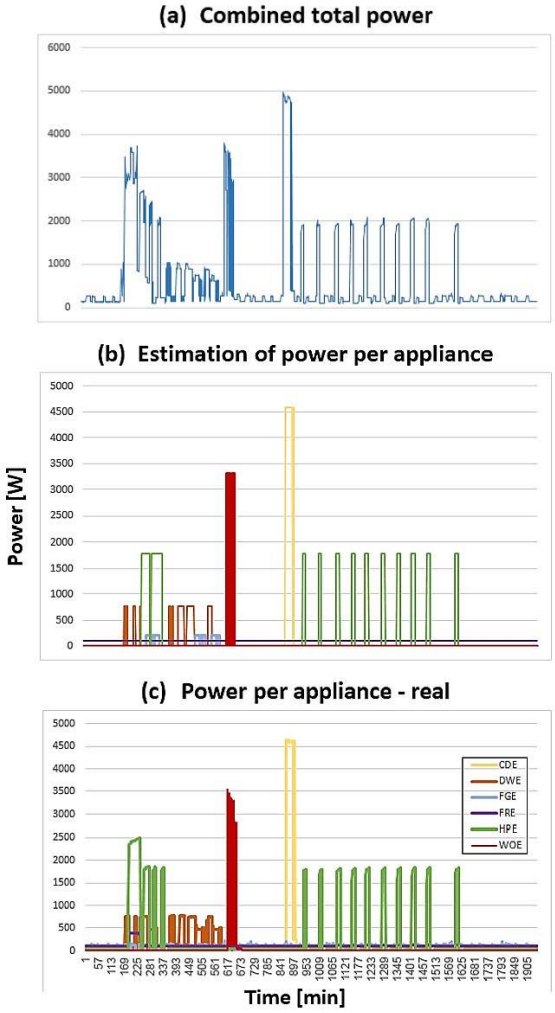

Figure 11. AMPds results for 32 h: (**a**) total power, (**b**) estimated disaggregated power consumption, and (**c**) true disaggregated power.

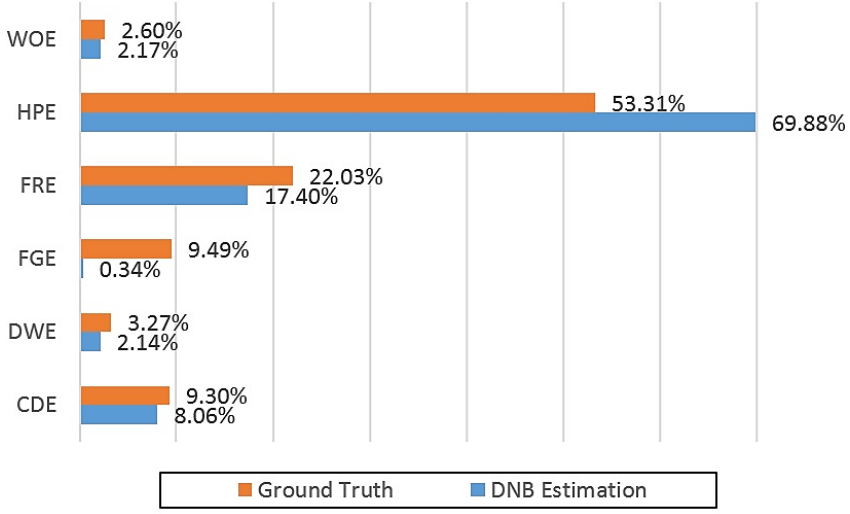

Figure 12. Bar graph showing the breakup of energy consumption per appliance for the ground truth versus estimated results by DNB.

6. Conclusions

This paper proposes a DNB technique to classify NILM events. NILM was formulated as a multi-dimensional problem, solved by the presented DNB process. DNB was examined in a limited-dimension space (*P*–*Q* plane) but showed that it can produce better results as more dimensions are applied. The presented method in this paper was extended for dealing with type II loads by the addition of watchdog timers. These timers guarantee that in the case of oscillating behavior of a load, the load will be treated as one on-off event rather than multiple events. This addition strengthens the ability of the presented approach to deal with a larger variety of loads.

In comparison with other methods, the DNB algorithm has shown an average improvement of over 30%, when using standard evaluation measures, while only requiring a single scan of the dataset. DNB is also easy to implement, as it is based on a simple clustering algorithm. The disadvantages of the presented method are the inherent order sensitivity of BIRCH and that it requires the extraction of several features from the smart meter. It is believed that order sensitivity can easily be handled, for example, by a preliminary observation of initial results in small scale samples, as demonstrated in Figure 11, or even by applying a preliminary learning period, although it would be redundant for most regular scenarios.

As future research, an expansion of DNB test scenarios is proposed, using additional datasets and appliances, in order to quantify the benefits of the use of various features. Further research and analysis of noise-effects is also suggested. The process itself could produce more accurate results with the addition of a dynamic technique for estimating power consumption separately for each identified event.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/en16073027/s1.

Author Contributions: Conceptualization, A.K. and Y.B.; formal analysis, A.K. and A.Y.; methodology, A.K., A.Y. and Y.B.; supervision, Y.B.; writing—original draft, A.K.; writing—review and editing, A.Y. and Y.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data is contained as Supplementary Material to the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yildiz, B.; Bilbao, J.I.; Dore, J.; Sproul, A.B. Recent advances in the analysis of residential electricity consumption and applications of smart meter data. *Appl. Energy* **2017**, *208*, 402–427. [CrossRef]
2. Dileep, G. A survey on smart grid technologies and applications, Renew. *Energy* **2020**, *146*, 2589–2625.
3. Winett, R.; Neale, M.; Grier, H. Effects of Self-Monitoring and Feedback on Residential Electricity Consumption. *J. Appl. Behav. Anal.* **1979**, *12*, 173–184. [CrossRef] [PubMed]
4. Sintov, N.D.; Schultz, P.W. Unlocking the potential of smart grid technologies with behavioral science Frontiers in Psychology. *Front. Psychol.* **2015**, *6*, 410. [CrossRef] [PubMed]
5. Zangheri, P.; Serrenho, T.; Bertoldi, P. Energy savings from feedback systems: A meta-studies' review. *Energies* **2019**, *12*, 3788. [CrossRef]
6. Arian, M.; Ameli, M.; Soleimani, V. Ghazalizadeh, Intelligent migration from smart metering to smart grid. In Proceedings of the 2011 IEEE Power Engineering and Automation Conference, Wuhan, China, 8–9 September 2011; Volume 2, pp. 547–552.
7. Kukuča, P.; Chrapčiak, I. From Smart Metering to Smart Grid Meas. *Science* **2016**, *16*, 142–148.
8. Walker, B.J. *Smart Grid System Report; 2018 Report to Congress, U.S.*; Dep. of Energy: Washington, DC, USA, 2018.
9. Giordano, V.; Gangale, F.; Fulli, G.; Jiménez, M.S. *Smart Grid Projects in Europe: Lessons Learned and Current Developments*; European Commission Joint Research Centre Institute for Energy: Luxembourg, 2013.
10. Hart, G.W.; Kern, E.C., Jr.; Schweppe, F.C. Non-intrusive appliance monitor apparatus. U.S. Patent 4,858,141, 15 August 1989.
11. Hart, G.W. Residential energy monitoring and computerized surveillance via utility power flows. *IEEE Technol. Soc. Mag.* **1989**, *8*, 12–16. [CrossRef]
12. Hart, G.W. Nonintrusive appliance load monitoring. *Proc. IEEE* **1992**, *80*, 1870–1891. [CrossRef]

13. Kim, H.; Marwah, M.; Arlitt, M.; Lyon, G.; Han, J. Unsupervised disaggregation of low frequency power measurements. In Proceedings of the 2011 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, Mesa, AZ, USA, 28–30 April 2011; pp. 28–30.
14. Wu, Z.; Wang, C.; Zhang, H.; Peng, W.; Liu, W. A time-efficient factorial hidden Semi-Markov model for non-intrusive load monitoring. *Electr. Power Syst. Res.* **2021**, *199*, 107372. [CrossRef]
15. Raiker, G.A.; Reddy, S.B.; Umanand, L.; Yadav, A.; Shaikh, M.M. Approach to non-intrusive load monitoring using factorial hidden markov model. In Proceedings of the 2018 IEEE 13th International Conference on Industrial and Information Systems (ICIIS), Rupnagar, India, 1–2 December 2018; pp. 381–386.
16. Dong, M.; Meira, P.C.M.; Xu, W.; Freitas, W. An event window-based load monitoring technique for smart meters. *IEEE Trans. Smart Grid* **2012**, *3*, 787–796. [CrossRef]
17. Jin-Gyeom, K.; Lee, B. Appliance classification by power signal analysis based on multi-feature combination multi-layer LSTM. *Energies* **2019**, *12*, 2804.
18. He, K.; Stankovic, L.; Liao, J.; Stankovic, V. Non-intrusive load disaggregation using graph signal processing. *IEEE Trans. Smart Grid* **2016**, *9*, 1739–1747. [CrossRef]
19. Wittmann, F.M.; López, J.C.; Rider, M.J. Nonintrusive Load Monitoring Algorithm Using Mixed-Integer Linear Programming. *IEEE Trans. Consum. Electron.* **2018**, *64*, 180–187. [CrossRef]
20. Machlev, R.; Levron, Y.; Beck, Y. Modified Cross Entropy method for classification events in NILM system. *IEEE Trans. Smart Grid* **2018**, *10*, 4962–4973. [CrossRef]
21. Machlev, R.; Tolkachov, D.; Levron, Y.; Beck, Y. Dimension reduction for NILM classification based on principle component analysis. *Elec. Power Sys. Res.* **2020**, *187*, 106459. [CrossRef]
22. Machlev, R.; Belikov, J.; Beck, Y.; Levron, Y. MO-NILM: A multi-objective evolutionary algorithm for NILM classification. *Energy Build.* **2019**, *199*, 134–144. [CrossRef]
23. Shunfu, L.; Zhao, L.; Li, F.; Liu, Q.; Li, D.; Fu, Y. A nonintrusive load identification method for residential applications based on quadratic programming. *Electr. Power Syst. Res.* **2016**, *133*, 241–248.
24. Samira, G.; Pamulapati, T.; Mallipeddi, R. Swarm and evolutionary algorithms for energy disaggregation: Challenges and prospects. *Int. J. Bio-Inspired Comput.* **2021**, *17*, 215–226.
25. Fahad, A.; Alshatri, N.; Tari, Z.; Alamri, A.; Khalil, I.; Zomaya, A.Y.; Bouras, A. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE Trans. Emerg. Top. Comput.* **2014**, *2*, 267–279. [CrossRef]
26. Ullah, A.; Haydarov, K.; Haq, I.U.; Muhammad, K.; Rho, S.; Lee, M.; Baik, S.W. Deep learning assisted buildings energy consumption profiling using smart meter data. *Sensors* **2020**, *20*, 873. [CrossRef]
27. Kwac, J.; Flora, J.; Rajagopal, R. Household energy consumption segmentation using hourly data. *IEEE Trans. Smart Grid* **2014**, *5*, 420–430. [CrossRef]
28. Lin, Y.; Tsai, M. Non-intrusive load monitoring by novel neuro-fuzzy classification considering uncertainties. *IEEE Trans. Smart Grid* **2014**, *5*, 2376–2384. [CrossRef]
29. Henao, N.; Agbossou, K.; Kelouwani, S.; Dubé, Y.; Fournier, M. Approach in nonintrusive type I load monitoring using subtractive clustering. *IEEE Trans. Smart Grid* **2015**, *99*, 1. [CrossRef]
30. Wang, A.; Longjun, B.; Chen, X.; Wang, C.G.; Hua, D. Non-intrusive load monitoring algorithm based on features of V–I trajectory. *Electr. Power Syst. Res.* **2018**, *157*, 134–144. [CrossRef]
31. Egarter, D.; Bhuvana, V.P.; Elmenreich, W. PALDi: Online load disaggregation via particle filtering. *IEEE Trans. Inst. Meas.* **2015**, *64*, 467–477. [CrossRef]
32. Qian, W.; Wang, F. Concatenate convolutional neural networks for non-intrusive load monitoring across complex background. *Energies* **2019**, *12*, 1572.
33. Min, X.; Wang, K.; Zhang, X.; Xu, Y. Non-intrusive load disaggregation based on deep dilated residual network. *Electr. Power Syst. Res.* **2019**, *170*, 277–285.
34. Kim, H.; Lim, S. Temporal patternization of power signatures for appliance classification in nilm. *Energies* **2021**, *10*, 2931. [CrossRef]
35. Fionn, M.; Legendre, P. Ward's hierarchical clustering method: Clustering criterion and agglomerative algorithm. *arXiv* **2011**, arXiv:1111.6285.
36. Said, B.K.; Streubel, R.; Yang, B. An approach for unsupervised non-intrusive load monitoring of residential appliances. In Proceedings of the 2nd International Workshop on Non-Intrusive Load Monitoring, Austin, TX, USA, 3 June 2014.
37. Chinthaka, D.; Makonin, S.; Bajić, I.V. Residential power forecasting using load identification and graph spectral clustering. *IEEE Trans. Circuits Syst. II: Express Briefs* **2019**, *66*, 1900–1904.
38. Bochao, Z.; Stankovic, L.; Stankovic, V. On a training-less solution for non-intrusive appliance load monitoring using graph signal processing. *IEEE Access* **2016**, *4*, 1784–1799.
39. Zhenyu, W.; Zheng, G. Residential appliances identification and monitoring by a nonintrusive method. *IEEE Trans. Smart Grid* **2011**, *3*, 80–92.
40. Jazizadeh, F.; Becerik-Gerber, B.; Berges, M.; Soibelman, L. An unsupervised hierarchical clustering-based heuristic algorithm for facilitated training of electricity consumption disaggregation systems. *Adv. Eng. Inform.* **2014**, *28*, 311–326. [CrossRef]

41. Zhang, T.; Ramakrishnan, R.; Livny, M. BIRCH: A new data clustering algorithm and its applications Data Min. *Knowl. Discov.* **1997**, *1*, 141–182. [CrossRef]
42. Wang, Y.; Chen, Q.; Hong, T.; Kang, C. Review of smart meter data analytics: Applications, methodologies, and challenges. *IEEE Trans. Smart Grid* **2019**, *10*, 3125–3148. [CrossRef]
43. Fontanini, A.D.; Abreu, J. A Data-Driven BIRCH Clustering Method for Extracting Typical Load Profiles for Big Data. In Proceedings of the 2018 IEEE Power & Energy Society General Meeting (PESGM), Portland, OR, USA, 5–10 August 2018; pp. 1–5.
44. Chaudhari, A.; Mulay, P. A bibliometric survey on incremental clustering algorithm for electricity smart meter data analysis. *Iran Jour. Comp. Sci.* **2019**, *2*, 197–206. [CrossRef]
45. Sun, L.; Zhou, K.; Yang, S. An ensemble clustering based framework for household load profiling and driven factors identification. *Sustain. Cities Soc.* **2020**, *53*, 101958. [CrossRef]
46. Ma, R.; Yu, N. A New Route for Energy Efficiency Diagnosis and Potential Analysis of Energy Consumption from Air-Conditioning System. In Proceedings of the 14th International Conference for Enhanced Building Operations (ICEBO2014), Beijing, China, 15–16 September 2014; pp. 14–17.
47. Makonin, S.; Popowich, F.; Bartram, L.; Gill, B.; Bajic, I.V. AMPds: A public dataset for load disaggregation and eco-feedback research. In Proceedings of the Electrical Power & Energy Conference (EPEC), Halifax, NS, Canada, 21–23 August 2013; pp. 1–6.
48. Ruano, A.; Hernandez, A.; Ruano, J.U.M.; Garcia, J. NILM Techniques for Intelligent Home Energy Management and Ambient Assisted Living: A Review. *Energies* **2019**, *12*, 2203. [CrossRef]
49. Nayyar, A.; Puri, V. Comprehensive Analysis & Performance Comparison of Clustering Algorithms for Big Data Review of Computer. *Eng. Res.* **2017**, *4*, 54–80.
50. Hripcsak, G.; Rothschild, A.S. Agreement, the f-measure, and reliability in information retrieval. *J. Amer. Med. Inform. Assoc.* **2005**, *12*, 296–298. [CrossRef]
51. Kolter, J.Z.; Johnson, M.J. REDD: A public data set for energy disaggregation research. In Proceedings of the Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA, USA, 24 August 2011; pp. 1–6.
52. Brown, M.; Bossley, K.M.; Mills, D.J.; Harris, C.J. High dimensional neurofuzzy systems: Overcoming the curse of dimensionality. In Proceedings of the 1995 IEEE International Conference on Fuzzy Systems, Yokohama, Japan, 20–24 March 1995; pp. 2139–2146. [CrossRef]
53. Cong, Y.; Liu, J.; Fan, B.; Zeng, P.; Yu, H.; Luo, J. Online Similarity Learning for Big Data with Overfitting. *IEEE Trans. Big Data* **2018**, *4*, 78–89. [CrossRef]
54. Zhu, X. *Knowledge Discovery and Data Mining: Challenges and Realities*; IGI Global: Hershey, PA, USA, 2007; pp. 118–119.
55. Samira, G.; Pamulapati, T.; Mallipeddi, R.; Lee, M. Energy disaggregation considering least square error and temporal sparsity: A multi-objective evolutionary approach. *Swarm Evol. Comput.* **2021**, *64*, 100909.
56. Adrian, F. Blued: A fully labeled public dataset for event-based nonintrusive load monitoring research. In Proceedings of the 2nd Workshop on Data Mining Applications in Sustainability (SustKDD), San Diego, CA, USA, 24 August 2011; Volume 2012.