The Journal of Supercomputing (2023) 79:14039–14062
https://doi.org/10.1007/s11227-023-05149-8

Thresholding methods in non-intrusive load monitoring

Daniel Precioso[1] · **David Gómez-Ullate**[2]

Accepted: 28 February 2023 / Published online: 1 April 2023

Abstract
Non-intrusive load monitoring (NILM) is the problem of predicting the status or consumption of individual domestic appliances only from the knowledge of the aggregated power load. NILM is often formulated as a classification (ON/OFF) problem for each device. However, the training datasets gathered by smart meters do not contain these labels, but only the electric consumption at every time interval. This paper addresses a fundamental methodological problem in how a NILM problem is posed, namely how the different possible thresholding methods lead to different classification problems. Standard datasets and NILM deep learning models are used to illustrate how the choice of thresholding method affects the output results. Some criteria that should be considered for the choice of such methods are also proposed. Finally, we propose a slight modification to current deep learning models for multi-tasking, i.e. tackling the classification and regression problems simultaneously. Transfer learning between both problems might improve performance on each of them.

Keywords Non-intrusive load monitoring (NILM) · Recurrent neural networks · Convolutional neural networks · Binary cross-entropy loss · mean squared error loss

Acronyms

AT	Activation-time thresholding
DL	Deep learning
MAE	Mean absolute error
ML	Machine learning
MP	Middle-point thresholding

✉ Daniel Precioso
daniel.precioso@uca.es

David Gómez-Ullate
david.gomezullate@ie.edu

[1] Department of Computer Science, Higher School of Engineering, Universidad de Cádiz, Av. Universidad de Cádiz, 10, 11519 Cádiz, Andalucía, Spain

[2] School of Science and Technology, IE University, Paseo de la Castellana, 259, 28046 Madrid, Madrid, Spain

NILM	Non-intrusive load monitoring
VS	Variance-sensitive thresholding

1 Introduction

Non-intrusive load monitoring (NILM) was proposed in 1992 by Hart [1] as a method to predict whether an individual appliance is active (ON) or inactive (OFF), by observing the total aggregated power load, having information of the nominal consumption of each appliance in each state. The first approach to NILM [1] employed Combinatorial Optimization, which at the time was the standard technique for disaggregation problems. For historical review of the evolution of NILM techniques, see for instance [2–4]. This first approach had a major shortcoming: the initial methods that employed combinatorial optimization performed the power disaggregation on each instant independently of the others, without considering the load evolution through time. These original algorithms were very sensitive to noise and were only accurate for houses with very few appliances. Thus, they could not be applied to real-life scenarios.

NILM algorithms received renewed attention at beginning of the twenty-first century, mostly thanks to the increased number of datasets coming from smart electric meters installed in domestic residences. These meters are able to record the power load from a household at short time intervals (one hour or less) and send those values to the electric company. The first open-source NILM datasets were published in 2011, and they triggered further research activity by setting benchmarks for models' comparison. These datasets stored high frequency time series for both aggregated and appliance power load [5–7]. For a recent comprehensive review on NILM datasets, see [8, 9]. Many of them are publicly available in the NILM-EU wiki.[1]

Soon after these datasets became available, the prevailing approach to NILM shifted from a combinatorial optimization approach to a supervised learning problem with times series in machine learning (ML) [10, 11]. Traditional ML methods such as hidden Markov models were initially used [12–14], while in recent years the incredible growth of deep learning (DL) algorithms has dominated the field [15]. Some popular architectures applied to NILM are recurrent neural networks, which are designed to process sequential data [16–19]. Convolutional neural networks, traditionally used for computer vision, have also been applied in NILM through the introduction of temporal poolings that facilitate the recognition of patterns at different time scales. These models have performed well even on datasets with low sampling rates [20–23]. Much of the recent effort of the NILM research community has been focused on improving the efficiency of estimation algorithms and computational speed, usually by presenting novel model architectures [17, 20] or even trying unconventional approaches such as finite state machines [24] and hybrid programming [25]. Unsupervised learning has also being proposed as a means to predict both the status and consumption

[1] http://wiki.nilm.eu/datasets.html.

of the monitored appliances [26–28]. These works first apply event detection to determine the time when an appliance changes state, called “ON/OFF events”, and then use *k*-means to cluster these events depending on the increase (decrease) of the aggregated load caused by the ON/OFF event. The ability of these unsupervised methods to associate the appliance consumption with its possible status is very relevant, as datasets used for supervised learning rarely include the real status, an issue that will be expanded further on this section.

The correlation between the state of the device switches and its consumption (as the former depends on the latter) has led some recent studies [29–31] to implement multi-task learning models. These algorithms are trained to predict both appliance status and power load at the same time. Learning both tasks simultaneously has been shown to improve performance due do transfer learning between the two tasks.

However, fewer works are devoted to investigating the formulation of the problem, and how this can affect the algorithm’s performance. For instance, uniformization of evaluation metrics for a better comparison of different models has been stressed by Pereira and Nunes [32], Makonin and Popowich [33], while dependence of accuracy on sampling frequency has been treated by Ruano et al. [34]. This becomes a relevant issue when trying to leverage theoretical NILM results on high frequency benchmark datasets to real-life scenarios where records are sampled at lower rates.

NILM datasets typically include both the aggregated power load and that of each monitored device, but not the appliance status (i.e. whether it is ON or OFF). Thus, a regression problem to predict the consumption of each device is naturally defined by the data. However, most works in NILM address the classification problem of determining whether the device is ON or OFF, rather than its consumption at each time interval. Defining a classification problem requires establishing a threshold or some procedure to determine the output categorical variable from the continuous output power load. Our main observation is that this process involves an external choice of thresholding method which is not included in the initial problem formulation. Depending on how this preprocessing step is performed, the performance and interpretation of the final results may vary in a significant manner. The main contribution of this paper is to highlight this matter and to discuss several possible ways to define a classification problem from the native regression problem. Moreover, interplay of both classification and regression problems might improve accuracy through multi-task models.

The paper is organized as follows: Sect. 2 formally introduces the necessary notation to define regression and classification problems for time series in supervised learning. Section 3 introduces three different thresholding methods. Two different deep learning (DL) models are presented and explained in Sect. 4. The purpose of studying two different models is to have more robust results and to ensure that the reported variations are not model dependent. The detailed methodology is carefully explained in Sect. 5, including data preprocessing, definition of training, validation and test sets, loss functions, optimization algorithms, and evaluation metrics. The results are exhibited in Sect. 6 for three monitored devices with different characteristics. Together with the results, we include a discussion on the criteria to choose the most convenient thresholding method. Section 7 shows a small modification to the usual DL architectures to optimize both classification and regression. Finally, concluding remarks and outline of open research problems are presented Sect. 8.

2 Problem formulation

Although several different formulations exist, NILM is often posed as a supervised learning problem, where the model is trained to take the aggregate power as input signal and predict the power or state (ON/OFF) of each monitored appliance. This power load is measured by the smart meter at a constant rate τ^*, which produces a series of power measurements[2] P_i at each sampled interval. For the analysis, it is often convenient to resample the original series at a larger sampling interval τ, which is part of the preprocessing step. For instance, in this paper, the native sampling interval for the UK-DALE dataset provided by the meters is $\tau^* = 6$ s, but the series has been resampled for the analysis at intervals of $\tau = 60$ s.

The aggregate power P_j at instant j is the sum over all appliances:

$$P_j = \sum_{\ell=1}^{L} P_j^{(\ell)} + e_j, \tag{1}$$

where L is the total number of appliances in the building, $P_j^{(\ell)}$ is the power of appliance ℓ at time j, and e_j is the unidentified residual load. All of these quantities are expressed in watts.

After resampling, the training set comprises a sequence of n_{tot} records labelled as $\{P_i\}_{i=0}^{n_{\text{tot}}}$. This series is split in chunks of size n that can be grouped in vectors as $\mathbf{P}_j = (P_{jn}, P_{jn+1}, \dots, P_{jn+n-1})$. As a result, we have a total of $n_{\text{train}} = n_{\text{tot}}/n$ such series, each of which will be an input to the model. The output of the model is sequences $\hat{\mathbf{P}}_j^{(\ell)}$ for each monitored appliance over the same time intervals. The pairs $\{\mathbf{P}_j, \mathbf{P}_j^{(\ell)}\}_{j=1}^{n_{\text{train}}}$ are considered as independent points in the training set.

Supervised learning problems are usually referred to as classification or regression problems depending on whether the output variables are categorical or continuous. In the NILM literature, both of these approaches have been considered in different contributions, but there has been hardly no works devoted to the interplay between both formulations. It is precisely this gap that we would like to fill with this analysis.

In the *regression approach*, the target quantities are the power load $\mathbf{P}_j^{(\ell)}$ for each device.

In the *classification approach*, the focus is on determining whether a given appliance is at time j in a number of possible states, typically ON or OFF. We assume therefore, for the sake of simplicity, that the appliance ℓ can be in one of two states at time j, which are $s_j^{(\ell)} = 0$ (OFF state) and $s_j^{(\ell)} = 1$ (ON state). It is not evident to

[2] Strictly speaking, smart meters measure the energy consumption during the interval $(t, t + \tau^*)$ divided by the length of the interval τ^*.

ascertain when a given appliance is ON or OFF by just looking at the power load. Thus, the usual criterion is to establish a threshold $\lambda^{(\ell)}$ for each appliance and define

$$s_j^{(\ell)} = H\big(P_j^{(\ell)} - \lambda^{(\ell)}\big), \tag{2}$$

where $H(x)$ denotes the Heaviside function

$$H(x) = \begin{cases} 1 & \text{if } x \geq 0, \\ 0 & \text{if } x < 0. \end{cases}$$

A correct definition of the classification approach thus involves a choice of threshold $\lambda^{(\ell)}$ for each appliance ℓ. A manual intervention to determine sensible threshold levels for each device over a large dataset would be unfeasible, plus it introduces a subjective element to the problem. Ideally, these thresholds should be determined by the series of data $\mathbf{P}_j^{(\ell)}$ alone, rather than being externally fixed by human intervention. Different algorithms to determine thresholds are examined below, which lead to rather different outcomes depending on the complexity of the input signal.

3 Thresholding

Given the input power signal, several different methods to set a threshold to determine the OFF and ON status of each appliance will be explored in this section.

3.1 Middle-point thresholding (MP)

In middle-point thresholding, the set of all power values from appliance ℓ in the training set $\{P_j^{(\ell)}\}_{j=1}^{n_{\text{tot}}}$ needs to be considered. A clustering algorithm is applied to split this set into two clusters which defines the centroid of each cluster. Typically, a k-means clustering algorithm can be applied for this purpose [35]. The two centroids for each class, after applying k-means, are denoted by $m_0^{(\ell)}$ (for the OFF state) and $m_1^{(\ell)}$ (for the ON state). In middle-point thresholding (MP), the threshold for appliance ℓ is fixed at the middle point between these values

$$\lambda^{(\ell)} = \frac{m_0^{(\ell)} + m_1^{(\ell)}}{2}. \tag{3}$$

3.2 Variance-sensitive thresholding (VS)

Variance-sensitive thresholding (VS) was recently proposed as a finer version of MP by Desai et al. [36]. It also employs k-means clustering to find the centroids for each class, but the determination of the threshold now takes into account not only the mean, but also the standard deviation $\sigma_k^{(\ell)}$ for the points in each cluster, according to the following formula

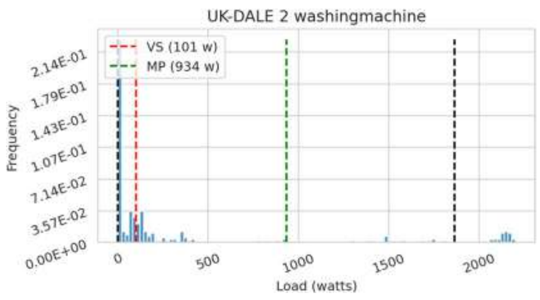

Fig. 1 Distribution of a washing machine power load through all the monitoring time. The records pertain to house 2 of UK-DALE dataset. The graph was cropped vertically as 0 W consumption is more than 17% of the total number of records. Black dashed lines mark the centroids found by k-means clustering (1 watts and 1866 watts), while color dashed lines are the thresholds fixed by different methods

$$\begin{aligned} d &= \frac{\sigma_0^{(\ell)}}{\sigma_0^{(\ell)} + \sigma_1^{(\ell)}} \\ \lambda^{(\ell)} &= (1-d)m_0^{(\ell)} + dm_1^{(\ell)} \end{aligned} \tag{4}$$

The motivation is that, if $\sigma_1 > \sigma_0$, then the threshold should move towards m_0 in order not to misclassify the points in class 1 that are further away from the centroid m_1. As a matter of fact, points in the OFF cluster usually have less variance, so the VS approach often sets its threshold lower than the MP approach. A comparison of both thresholding methods on a specific set of power measurements can be seen in Fig. 1. Note that VS defaults to MP precisely when $\sigma_0 = \sigma_1$, i.e. when both clusters have the same variance.

3.3 Activation-time thresholding

The last two methods only use data from the distribution of power measurements in order to fix the threshold for an appliance. It often happens that due to noise in the smart meters or devices, some measurements during short time intervals are either absent while the device is operating, or produce abnormal peaks during the OFF state. For this reason, to ensure a smoother behavior, Kelly and Knottenbelt [37] set both a power threshold and a time threshold. The power threshold could be fixed by MP or VS or fixed externally by hand as done in [37]. The time threshold $(\mu_0^{(\ell)}, \mu_1^{(\ell)})$ specifies the minimum length of time that device ℓ must be in a given state, e.g. if a sequence of power measurements are below $\lambda^{(\ell)}$ for a time $t < \mu_0^{(\ell)}$, then that sequence is considered to be in the previous state (ON in this binary case).

In [37], both power and time thresholds are chosen empirically, after analyzing the appliance behavior. Table 1 shows the values of the thresholds relevant to our work. The threshold λ is chosen usually at lower values, as the time threshold already filters noisy records. It would be desirable to turn this thresholding method into a fully automated data driven algorithm, in order to remove all subjective inputs.

Table 1 Activation-time thresholding (AT) values used in this work

Threshold	Dishwasher	Fridge	Washing machine
λ (W)	10	50	20
μ_0 (s)	30	1	3
μ_1 (s)	30	1	30

λ is the threshold that separates the consumption of ON and OFF states, μ_0 (μ_1) is the time that the appliance consumption must spend below (above) the threshold to be considered OFF (ON)

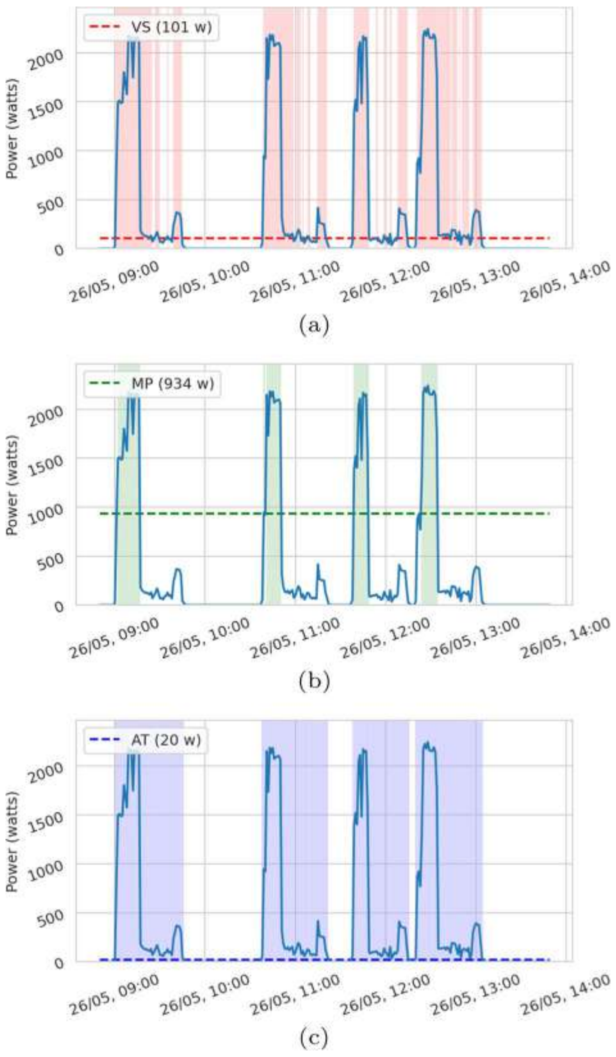

Fig. 2 Sample from the washing machine power load sequence, depicting how different thresholding methods classify each instance as ON or OFF. ON states are highlighted. **a** Variance-sensitive thresholding (VS). **b** Middle-point thresholding (MP). **c** Activation-time thresholding (AT)

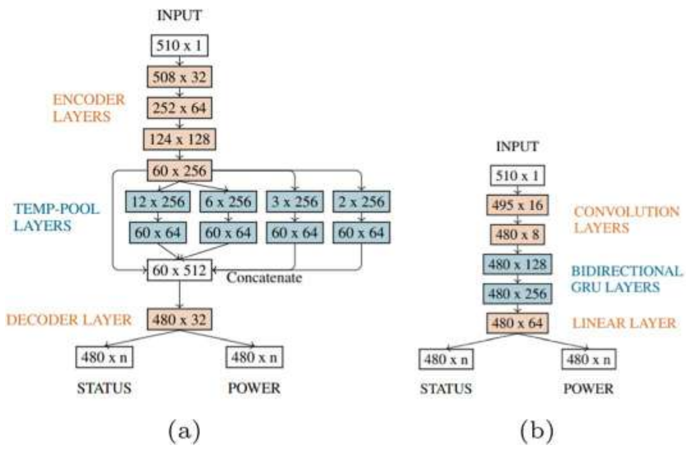

Fig. 3 Architecture of each DL model. There are two possible outputs (appliance status or power consumption), the model trains differently depending of which output we chose. Models are as follows: **a** CONV. **b** GRU

Figure 2 compares the three thresholding methods. Each graph shows the three values of $\lambda^{(\ell)}$ for a given device, together with the result of applying each thresholding method to the same input series. Observe that the same power data give rise to rather different series for the ON/OFF status depending on the choice of thresholding method. We thus see that there are multiple ways to define a classification problem given the input signal.

A comparison and a discussion of each thresholding method after training state-of-the-art NILM for regression and classification problems will be performed in Sect. 6.

4 Neural networks

Almost all state-of-the-art models propagate their inputs through one or more convolutional layers [38, 39]. This is done to ensure that the models are translation invariant. As NILM is related to time series, many studies also add recurrent layers (e.g. LSTM or GRU) to their networks [16, 18]. These layers tend to get very good results on sequence-related problems. In this work, two different models are considered: one that relies purely on convolutions, and other that also applies recurrent layers after the convolutions.

It is important to stress that both of these neural networks can be applied to train a classification model or a regression model, the only difference in their architecture lies in the last layer, where an additional softmax layer needs to be added for the classification problem.

4.1 Convolutional network

The first model (CONV) includes as feature extractor only convolutional layers, inspired on the architecture from the work of Massidda et al. [21]. The general scheme follows the classic approach to the semantic segmentation of images in computer vision. See Fig. 3a to better understand the following model explanation.

The CONV model receives as input a vector with size $L_{\text{in}} = 510$ which represents the household aggregated power over an $8\frac{1}{2}$ hour interval. The vector is propagated through an encoder, characterized by alternating convolution and pooling modules. Each encoder layer begins with a convolution of kernel size 3 and no padding, then applies batch normalization and ReLU activation, and ends with a max pooling of kernel size 2 and stride 2. Only the last layer omits the max pooling step. Encoder layers increase the space of the features of the signal at the cost of decreasing the temporal resolution.

After that, the *Temporal Pooling* module aggregates the features at different resolutions, which is reminiscent of inception networks [40]. Four different average poolings are applied, with kernel sizes 5, 10, 20 and 30; having the same stride as kernel size. Each of those layers then propagate their values through convolution layers of kernel size 1 and no padding, followed by a batch normalization and ReLU activation. All of their outputs are then concatenated.

Finally, the decoder module applies one convolution of kernel size 8 and stride 8, followed by batch normalization. It then bifurcates into two different outputs: the appliance status and the appliance power. Both outputs are computed by propagating the network values through one last convolution layer of kernel size 1 and padding 1. In the case of status classification, a softmax activation layer is further applied. Both status and power load output vectors have the same sampling frequency as the input aggregate load, but they have a shorter length $L_{\text{out}} = 480$ as explained in Sect. 5.

4.2 Bidirectional GRU network

Some authors tend to connect convolutional and recurrent layers to extract temporal correlations out of the input sequence [17, 37]. This second model follows a prototypical GRU scheme, depicted in Fig. 3b. The input and output layers are the same as in the previous model. For the processing units, the GRU model propagates the input vector through two convolutional layers with kernel size 20, padding 2 and stride 1, before applying the recurrent (bidirectional GRU) layers.

One can see that the model architecture is rather lightweight. However, GRU takes longer to train than CONV: adapting the GRU weights requires a lot of computation, compared to updating the weights of convolutional layers.

Table 2 UK-DALE dataset features

Date of release	2014
Location	United Kingdom
Number of households	5
Meter units	Watts
Sampling frequency	6 s

Table 3 UK-DALE dataset contents

Building	1	2	3	4	5
Total time (days)	> 365	235	39	206	137
Appliances	53	18	4	11	24

5 Methodology

5.1 Preprocessing

In order to make our results reproducible and allow them to compare with other works, the analysis focuses on the UK-DALE dataset [7], which is a standard benchmark for NILM.

Only houses 1, 2 and 5 have been used for this work. The target appliances used in the analysis, which are found in the three buildings, are as follows: fridge, dishwasher and washing machine. This choice of houses and appliances is common in other works [21, 37, 41], as they seek to monitor appliances with distinguishable load absorption patterns and relevant contribution to the total power consumption (Tables 2 and 3).

Every power load series was down-sampled from 6 s to 1 min-frequency, taking the average value of the power in the time intervals. It should be stressed that sub-sampling the input series might cause relevant information loss, as perhaps high frequency power variations could be strongly associated with the identification of a given device. The devices chosen in this study have high power loads and large operation times, so this sub-sampling will likely have a small effect in model performance for the particular choice of devices under study. The main reason to down-sample the input data, in accordance with Massidda et al. [21] and other authors, is to train a model under less favorable, but more widely applicable and realistic conditions, as NILM datasets often have a lower frequency.

After this down-sampling, every input sequence comprises $8\frac{1}{2}$ hours of time, which amount to $L_{\text{in}} = 510$ records. Since the models use convolutions with no padding, the first and last records of each series are dropped in the output, thus leading to an output sequence having $L_{\text{out}} = 480$ records, i.e. 8 h. We have divided the original time series into input sequences with an overlap of 30 records between consecutive input sequences, so that the output sequence are continuous in time and have no gaps. Aggregate power load is normalized, dividing the load by a reference power value of 2000 W for numerical stability. Each input series is

Table 4 Size of training, validation and test sets

	Train	Validation	Test
Number of points (N)	1941	183	183
Total time (days)	687	65	65

Table 5 Fraction of activation time (in %) for each device over the train and test sets, i.e. the proportion of time that each appliance in operating (is in its ON state)

Threshold	Set	Dishwasher	Fridge	Washing machine
MP	Train	0.77	42	0.97
	Test	0.62	46	1.7
VS	Train	0.83	44	1.8
	Test	0.67	47	2.5
AT	Train	2.3	42	4.6
	Test	2.4	45	7.3

further normalized by subtracting its mean. Thus, the following input and output series for the regression problem are defined

$$\text{Regression:} \qquad \boldsymbol{x}_j = \frac{\boldsymbol{P}_j - \bar{P}_j}{2000}, \qquad \boldsymbol{y}_j^{(\ell)} = \frac{\boldsymbol{P}_j^{(\ell)}}{2000}, \tag{5}$$

where $\bar{P}_j = \frac{1}{L_{\text{in}}} \sum_{i=1}^{L_{\text{in}}} P_{j,i}$.

To define the classification problem, the target $\boldsymbol{y}_j^{(\ell)}$ is the appliance status which is computed from $\mathbf{P}_j^{(\ell)}$ using the thresholding methods described in Sect. 3. More specifically, we have

$$\text{Classification:} \qquad \boldsymbol{x}_j = \frac{\boldsymbol{P}_j - \bar{P}_j}{2000}, \qquad \boldsymbol{y}_j^{(\ell)} = \boldsymbol{s}_j^{(\ell)}, \tag{6}$$

where $\boldsymbol{s}_j^{(\ell)}$ is a series of binary values defined by (2).

The training set for each problem is built by adding the first 80% sequences from each of the three buildings, which amounts to 1941 sequences (describing a total time of 687 days of measurements). The validation set is built by using the subsequent 10% records from house UK-DALE 1, for a total of 183 sequences (65 days), while the test set is composed of the last 10% sequences of the same building, having the same size as the validation set. This is summarized in Table 4.

To assess whether the train-test split is balanced, it will be useful to report how often a given device has been ON during the training and test sets, depending on which thresholding method has been applied. The results can be seen in Table 5.

It is important to stress a number of things from the observation of this table. First, the fraction of ON states is clearly dependent on the thresholding method, as it was already clear from Fig. 2. Next, dishwasher and washing machine are

Table 6 Fraction of time (%) where fridge, dish washer and washing machine were simultaneously active

Threshold	Set	Number of active devices			
		0	1	2	3
MP	Train	56.76	42.50	0.73	0.00
	Test	52.95	45.98	1.07	0.00
VS	Train	55.25	43.61	1.14	0.00
	Test	51.30	47.15	1.55	0.00
AT	Train	54.08	42.99	2.87	0.05
	Test	49.34	46.05	4.58	0.04

only sparsely activated, while the fridge is considered to be ON roughly half of the time. In these two cases, we are dealing with an imbalanced class problem, which should be taken into account when defining and interpreting the appropriate metrics. Finally, in all cases but specially for the washing machine, the prevalence of the positive class differs greatly from the training to the test set. This of course could happen because these periods have been chosen consecutive in time (in accordance with other works).

Another relevant problem in the NILM research is disaggregation accuracy when there is more than one appliance operating in the same time window. Once again, this problem is tied to the choice of a thresholding method, as it determines the operating times of each appliance and consequently the possible overlaps. Table 6 shows the frequency at which one or more of the devices were active simultaneously, depending on the thresholding method applied. MP produces the least amount of overlaps accounting for roughly 1% of the total time, while AT produces the most, exceeding a 3% of the time. In addition, AT is the only thresholding method that finds the three appliances operating at the same time, partly because this is the method that produces the highest amount of ON states in general.

5.2 Training

Each of the models described in Sect. 4 was trained for 300 epochs. Training data were fed to the model in batches of 32 sequences, shuffled randomly.

The loss function for the regression problem is the mean square error or L2 metric, given by

$$\mathcal{L}_{\text{reg}}^{(\ell)} = \frac{1}{N_{\text{train}}} \sum_{j=1}^{N_{\text{train}}} \frac{1}{L_{\text{out}}} \sum_{i=1}^{L_{\text{out}}} \left(y_{j,i}^{(\ell)} - \hat{y}_{j,i}^{(\ell)} \right)^2 . \tag{7}$$

The standard choice of loss function for the classification problem is binary cross-entropy:

Table 7 Mean absolute error (MAE) (in watts) for regression models on each appliance

Model	Dishwasher	Fridge	Washing machine
CONV	11.6	27.0	18.3
GRU	8.1	28.7	15.0

$$\mathcal{L}^{(\ell)}_{\text{class}} = \frac{1}{N_{\text{train}}} \sum_{j=1}^{N_{\text{train}}} \frac{1}{L_{\text{out}}} \sum_{i=1}^{L_{\text{out}}} \left(y^{(\ell)}_{j,i} \cdot \log \hat{y}^{(\ell)}_{j,i} + (1 - y^{(\ell)}_{j,i}) \cdot \log(1 - \hat{y}^{(\ell)}_{j,i}) \right), \tag{8}$$

where $\hat{\boldsymbol{y}}^{(\ell)}_j$ is the probability that device ℓ is ON at time step j.

During the 300 training epochs, we keep the model that achieves the minimum loss over the validation set, using an Adam optimizer for weights update, with a starting learning rate of 10^{-4}.

Both data preprocessing and neural network training were performed on Python. Specifically, the models were written on Pytorch and trained in a GPU NVidia GeForce GTX 1080 with 8 GB of VRAM, NVIDIA-SMI 440.95.01 and CUDA v10.2. The code for this paper is available online[3] and the data come from a public dataset, so all results reported in this paper are reproducible. Using the configuration stated in this section, CONV models took from 7 to 8 min to train 300 epochs, while GRU architectures took 16 min. It is instructive to analyze how would these computational times scale with the number of monitored devices. For a different number of devices, both DL architectures stay the same except for the output layer, and computational complexity can be shown to scale linearly with this number. When training CONV (GRU) models with two output appliances, we saw an average decrease of 4% (3%) on training time, and a 8% (6%) decrease when training with just one device. An estimation of training time of both models for n output devices ($2 \leq n \leq 20$) would be:

$$\text{CONV Training time} \approx 6 \times (1 + 0.04\, n) \text{ min}$$
$$\text{GRU Training time} \approx 15 \times (1 + 0.03\, n) \text{ min}$$

In this case, adding or subtracting devices did not have a significant impact on the overall performance, which is probably due to the fact that there are not many overlaps among them (see Table 6).

5.3 Metrics

Although metrics are clearly related to loss functions used for model training, the main difference is that the reported metrics are not required to be differentiable.

[3] https://github.com/UCA-Datalab/nilm-thresholding.

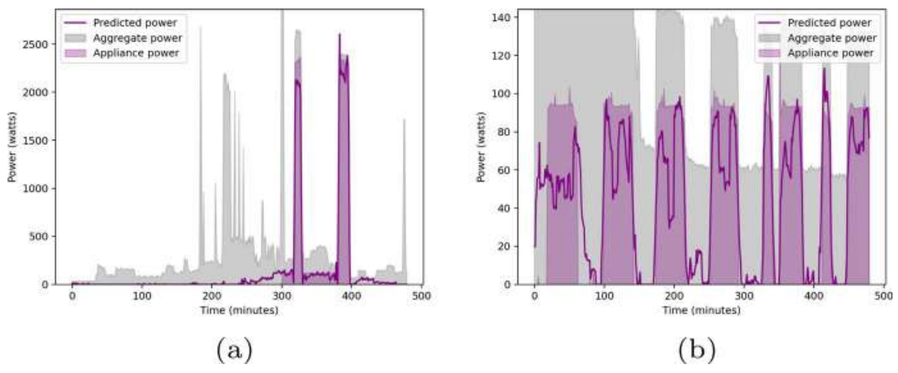

Fig. 4 Output of the CONV regression model. Aggregated power load (input signal) is shown in gray, the real consumption of the device is represented by a light purple shadow, while the predicted load is depicted by a dark purple line. Devices are: **a** dishwasher and **b** fridge

Table 8 F_1-scores for classification models on each appliance and threshold

Threshold	Model	Dishwasher	Fridge	Washing machine
MP	CONV	0.93	0.87	0.93
	GRU	0.84	0.87	0.87
VS	CONV	0.93	0.87	0.88
	GRU	0.84	0.87	0.82
AT	CONV	0.91	0.86	0.97
	GRU	0.90	0.86	0.96

When the output is a continuous variable, the chosen metric is the L1 error or MAE rather than the Root Mean Squared Error, since the latter tends to give too much importance to large deviations.

When the output variable is categorical, the F_1-score is an appropriate metric that combines precision and recall in balanced problems where there is no preference to achieve a better classification of the ON or OFF classes.

6 Results

6.1 Regression problem

The metrics obtained over the test set by the two models (CONV and GRU) in the regression problem for each of the three appliances $\mathscr{E} = \{\text{dishwasher, fridge, washing machine}\}$ are shown below (Table 7).

To understand the complexity of disaggregating the power signal in NILM, a typical time series from the test set has been plotted in Fig. 4. The plot shows the

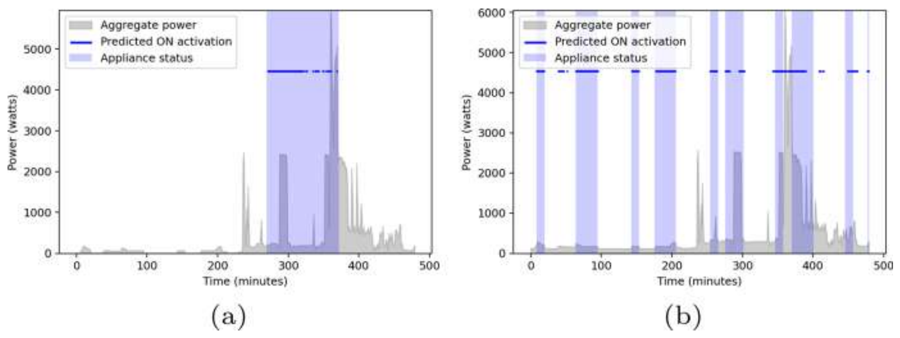

Fig. 5 Output of the CONV classification model applying activation-time thresholding (AT). Aggregated power load (input signal) is shown in gray and the ON status (derived from AT on the real consumption) is highlighted with a light blue shadow, while the predicted activations (ON states) are depicted by dark blue dots. Devices are **a** dishwasher and **b** fridge

input signal $\mathbf{P}_j$ (aggregated power load), the real power $\mathbf{P}_j^{(\ell)}$ and the predicted power $\hat{\mathbf{P}}_j^{(\ell)}$ obtained from the CONV model for dishwasher and fridge.

In the first graph of Fig. 4, the model has identified correctly the two main power peaks of 2200 W corresponding to the dish washer, but has properly ignored other similar large peaks occurring earlier in the series. In the second graph, we observe that the fridge power estimation is often masked by the presence of other devices, having a mean value of just 100 W but a very periodic activation pattern. When the aggregate power load is small, the model is able to better resolve the signal coming from the fridge.

6.2 Classification problem

As mentioned above, the classification problem is not uniquely defined since the raw data do not include the real intervals where each device was ON/OFF, but only its consumption. Thus, three different classification problems must be considered depending on the choice of thresholding method described in Sect. 3. For each possible value of (model, thresholding, device), the F_1-score over the test set is reported in Table 8.

The next to last line shows that our results are in good agreement with those reported by Massidda et al. [21]. Also, the results show that CONV has a slightly better F_1-score that GRU for the classification problem in all three devices and thresholding methods, although both of them show a very good performance (see Fig. 5). In general, the classification problem for the fridge is harder, for reasons that have been already mentioned above.

It is also instructive to represent the input signal $\mathbf{P}_j$, together with the real output signal $\mathbf{P}_j^{(\ell)}$ describing the status of device ℓ and the predicted status $\mathbf{s}_j^{(\ell)}$, to grasp the nature of NILM problem for classification. Figure 5 shows the output of CONV on a given series of records from the test set where the three devices have been ON

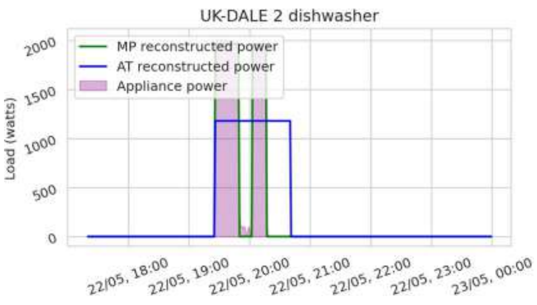

Fig. 6 Device power load reconstruction from two different thresholding methods: middle-point thresholding (MP) in green and activation-time thresholding (AT) in blue. The real consumption of the device is shown in purple

Table 9 MAE scores (in watts) for classification models and intrinsic error after reconstructing the power load

Threshold	Model	Dishwasher	Fridge	Washing machine
MP	Intrinsic	3.5	4.7	4.0
	CONV	9.1	21.5	39.5
	GRU	11.3	21.8	41.7
VS	Intrinsic	4.4	4.8	6.6
	CONV	9.3	21.0	40.8
	GRU	11.6	22.2	41.9
AT	Intrinsic	26.4	4.7	7.4
	CONV	19.8	21.3	57.3
	GRU	21.1	22.5	56.4

(sometimes simultaneously). Observe that the model is able to discriminate, with a very good precision, the periods in which each of the devices are activated, just from the observation of the aggregated power signal.

6.3 Reconstructing the power signal

A very natural question to address is which of the three proposed thresholding methods should be preferred. One would naively think that the one leading to a better F_1 -score would be the best choice, if this was only based on prediction performance. However, placing a trivial threshold of zero would yield state ON for all time intervals at the training and test sets, and any decent ML method would immediately learn this, thus reaching a perfect F_1-score but having no useful interpretation at all. Thus, prediction performance should be balanced with a way of judging which method is more meaningful. In the absence of any other external information on when each device can be considered to be ON/OFF, an objective quantitative argument to tackle this question must be found.

For this purpose, we propose to reconstruct the power signal from each device and compare the reconstructed signal with the original power load of the device. We compute the average power $\bar{P}_{\mathrm{ON}}$, (resp. $\bar{P}_{\mathrm{OFF}}$) for device ℓ during the periods that are considered to be ON, (resp. OFF) after applying the thresholding method, and reconstruct the power series with these binary values.

More specifically, we have

$$\bar{P}^{(\ell)}_{\mathrm{ON}} = \frac{1}{N_{\mathrm{train}}} \sum_{j=1}^{N_{\mathrm{train}}} \frac{1}{L_{\mathrm{out}}} \sum_{i=1}^{L_{\mathrm{out}}} s^{(\ell)}_{j,i} P_{j,i}, \tag{9}$$

$$\bar{P}^{(\ell)}_{\mathrm{OFF}} = \frac{1}{N_{\mathrm{train}}} \sum_{j=1}^{N_{\mathrm{train}}} \frac{1}{L_{\mathrm{out}}} \sum_{i=1}^{L_{\mathrm{out}}} \left(1 - s^{(\ell)}_{j,i}\right) P_{j,i}, \tag{10}$$

and reconstruct a binary power load for device ℓ as

$$\mathbf{BP}^{(\ell)}_{j} = \bar{P}^{(\ell)}_{\mathrm{ON}} \mathbf{s}^{(\ell)}_{j} + \bar{P}^{(\ell)}_{\mathrm{OFF}} \left(1 - \mathbf{s}^{(\ell)}_{j}\right) \tag{11}$$

The reconstructed power series can be seen, together with the original series in Fig. 6 for two of the thresholding methods, corresponding to the same data as Fig. 2.

The *intrinsic error* between the original $\mathbf{P}^{(\ell)}$ and reconstructed series $\mathbf{BP}^{(\ell)}$ can be defined as the MAE over the training set. The name *intrinsic* is justified because this error is prior to any prediction method. The results for the three devices and thresholding methods are shown in Table 9.

From this comparison, it follows that the activation-time thresholding (AT) is the one having the largest intrinsic error, while MP offers the closest reconstructed power series. The fridge has similar intrinsic error for all three methods since the original series is very regular, being almost a binary series itself (see Fig. 4).

As we mentioned above, it is not enough to look only at the classification metrics in Table 8 to judge which is the best thresholding method for a NILM problem. For this reason, given the estimation output of the classification problem, we compute the reconstructed binary series and compare it with the original power series. MAE averaged over the test set are reported in Table 9 for the two models (GRU and CONV).

The first thing to note is that naturally these metrics are generally larger than the intrinsic errors, as they incorporate the errors in the classification.

As shown in previous tables, CONV has a slightly better performance than GRU. It is also worth noting that the MAE for the washing machine has increased by a factor of 10 with respect to the intrinsic error, while in the other two devices, the factor is close to 4. The most likely explanation for this deviation is due to the train-test splitting: most of the error comes from activation periods, and these occur twice more often in the test than in the training set for the washing machine (see Table 5) while there is hardly no variation in the other two devices. This brings back the already mentioned remark on the importance of having a train-test split that preserves the distribution of classes.

Finally, these MAE values should be compared with the ones obtained by training for a pure regression problem (see Table 7). We observe that the results are comparable, and for the fridge they are even better in the reconstructed case. The explanation comes from the fact that the raw power signal for the fridge is almost binary, so the reconstructed signal matches this behavior properly and the ON/OFF values $\bar{P}_{\mathrm{ON}}$ and $\bar{P}_{\mathrm{OFF}}$ calculated over the training set are very close to the real values. Thus, in this case good metrics for the classification problem immediately translate into good

Table 10 F_1-scores for regression models after thresholding

Threshold	Model	Dishwasher	Fridge	Washing machine
MP	CONV	0.89	0.77	0.93
	GRU	0.90	0.74	0.91
VS	CONV	0.77	0.78	0.81
	GRU	0.81	0.75	0.83
AT	CONV	0.09	0.76	0.46
	GRU	0.27	0.72	0.65

scores for the reconstructed series. By contrast, addressing the regression problem directly is harder for the fridge, where the regression curve often fails to reconstruct this signal, specially when it is masked by larger signals coming from other devices (see Fig. 4b).

6.4 Classification metrics on the regression problem

In the last section, models were trained for classification, but the MAE was evaluated for the reconstructed power signal and compared with the same metrics obtained by directly training the model for regression. In this section, the opposite procedure will be tackled: thresholding methods are applied on the real and predicted power signal obtained from the output of the regression model, and the F_1 metric is then computed over these binary series. The results can be seen in Table 10.

These scores are on average worse than the ones from the original classification approach (Table 8). In particular, F_1-scores of AT for dishwasher and washing machine are extremely low, which is caused by the small power threshold set by the thresholding formulation (see Table 1). During periods of inactivity, regression models output values that, although being relatively small compared to the power peaks of dishwasher and washing machine, are high enough to surpass the AT power threshold, thus triggering the ON state and causing many false positives.

7 Balancing classification and regression

This section explores a slight modification of the deep learning (DL) models introduced in Sect. 4, to address a multi-task problem, i.e. to tackle the regression and classification problem at the same time. The network architectures contain two different heads, one for regression and the other for classification, but the whole feature extracting base is common to both. A weighted combined loss function can be defined that balances the loss of the two output layers in the following manner:

$$\mathcal{L}^{(\ell)}_{\text{tot}} = w \cdot \mathcal{L}^{(\ell)}_{\text{class}} + (1 - w) \cdot \mathcal{L}^{(\ell)}_{\text{reg}}/k, \tag{12}$$

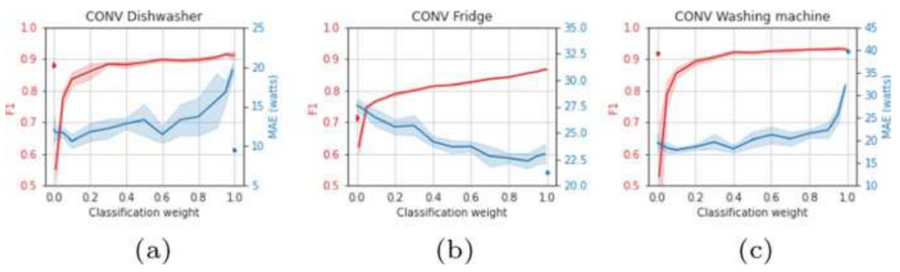

Fig. 7 Scores of the CONV model using middle-point thresholding (MP), trained for both regression and classification outputs, depending on the relative weight given to classification. Appliances are **a** dishwasher, **b** fridge and **c** washing machine

where $\mathcal{L}^{(\ell)}_{\text{class}}$ is the binary cross-entropy (8), $\mathcal{L}^{(\ell)}_{\text{reg}}$ is the Mean Squared Error (7), and k is a constant to normalize both losses so that they have comparable magnitude, predicted to be $k = 0.0066$. The constant $w \in [0, 1]$ allows to shift between pure classification and regression. The purpose of this combined training is to allow for transfer learning between the regression and classification problem that, although strongly related, are indeed different ML tasks. Both models have been trained using different values of the weight w with MP thresholding (other methods showed a similar behavior). For each value of w, the model is trained five times with random weight initializations, as explained in Sect. 5. We show the output metrics MAE and F_1-score for varying w in the figures below. Only CONV model is displayed, as both models behaved similarly. The advantage of using a combined loss function for the NILM classification problem is clear by looking at Fig. 7.

Note that when $w = 0$, the model does not train for classification. For this reason, we include for $w = 0$ a single point for the F_1 curve, corresponding to applying the thresholding method on the regression output, as we did in Sect. 6.4. Likewise, for $w = 1$, the model does not train for regression. Given the absence of a regression score for $w = 1$, we employed the MAE obtained by reconstructing the power signal from the classification output, as explained in Sect. 6.3.

Looking at the results in Fig. 7, we observe a very different behavior for the fridge than for the other two devices, due to their different characteristics already mentioned. In the dishwasher and washing machine, the F_1-score grows monotonically with w. Likewise, the MAE in both models and devices tends to grow for larger w, which is natural since the model has a smaller weight for the regression problem. As for the extra points in the graphs, for the dishwasher, we see that the F_1-score obtained by thresholding a pure regression output (red dot) is comparable to the best score obtained by larger weights in classification. For the fridge, the behavior is different from the other two devices. While the F_1-score behaves similarly, the MAE for regression decreases with w, which is clearly counter-intuitive: the model prioritizes classification loss, and in doing so it performs better in the regression problem as well. Also, the purely reconstructed signal for $w = 1$ (blue dot) has a better MAE than any of the models trained for regression. This fact can again be explained by the second graph in Fig. 4: the regular (almost binary) activation pattern of the fridge is much better captured by a classification model with the right threshold than by a regression model, since the weak signal of the fridge is often masked by that of other devices.

In summary, the advantage of using a combined multi-task approach over a single classification/regression approach is seen to be device dependent. In the three devices considered, only for the fridge the single regression task was improved by considering also the classification problem. Further study is needed to ascertain which types of devices will improve in their NILM performance with combined loss functions.

8 Conclusions and future research

Non-intrusive load monitoring is typically framed as a classification problem, where the input data are the aggregated power load of the household and the output data are the sequence of ON/OFF states of a given monitored device. Creating a classification problem from the raw power signal data requires an external determination of the status by some thresholding method. Three such methods are discussed in Sect. 3 and how they lead to classification problems with different results. A discussion of what is the most appropriate method should not be based on the performance achieved by predictive models alone, but include also some objective way to judge the interpretability of the results. An objective criterion suggested in the paper is to use the intrinsic error, i.e. MAE between the original power series and reconstructed binary series.

Deep learning (DL) models can be trained to minimize the regression loss (7) or the classification loss (8), but it is also possible to combine both into a weighted loss introducing an extra hyperparameter. This parameter balances the weight given to both problems, that are effectively solved both at a time. The optimal choice of this parameter depends on the characteristics of the device.

This work can possibly be extended in a number of ways that the authors plan to tackle in the future:

- Thresholding methods should be extended to multi-state NILM, where each device can operate on a finite number $k \geq 2$ of states. Automatic determination of the best number of states should be feasible from the power series by using hierarchical clustering or the elbow method. The intrinsic method described in the paper will generalize naturally to multi-state NILM classification.
- Thresholding methods should be entirely algorithmic. Two of them (MP and VS) already are but AT needs some parameters to be externally fixed. These free parameters, the time thresholds $(\mu_0^{(\ell)}, \mu_1^{(\ell)})$ for each device, should rather be automatically determined to minimize the intrinsic error defined in Sect. 6.3.
- NILM requires to address chronological versus random splitting of records to form the training, validation and test sets. Similar issues are key in discussing fraud detection methods, where fraud techniques evolve in time and differ chronologically throughout the time span of the dataset.

- The study should be extended to larger datasets like Pecan Street [42] or ECO [43], addressing also the generalization capacity of DL models to cope with unseen devices not present in the training set.

Acknowledgements DP gratefully acknowledges an Industrial PhD grant from Universidad de Cádiz. The research of DGU is supported in part by the Spanish Agencia Estatal de Investigación under grants PID2021-122154NB-I00 and TED2021-129455B-I00, and by a 2021 BBVA Foundation project for research in Mathematics. He also acknowledges support from the EU under the 2014-2020 ERDF Operational Programme and the Department of Economy, Knowledge, Business and University of the Regional Government of Andalusia (FEDER-UCA18-108393).

Author Contributions DP write the code used in this project, under the supervision of DG-U. Both Daniel Precioso and DG-U contributed equally to the manuscript.

Funding Funding for open access publishing: Universidad de Cádiz/CBUA. This research has been financed in part by the Spanish Agencia Estatal de Investigación under grants PID2021-122154NB-I00 and TED2021-129455B-I00, and by a 2021 BBVA Foundation project for research in Mathematics. He also acknowledges support from the EU under the 2014-2020 ERDF Operational Programme and the Department of Economy, Knowledge, Business and University of the Regional Government of Andalusia (FEDER-UCA18-108393).

Data availability The dataset used in this study comes from Kelly and Knottenbelt [7] and is publicly available in the following link: https://data.ukedc.rl.ac.uk/browse/edc/efficiency/residential/EnergyConsumption/Domestic/UK-DALE-2015.

Code Availability The code developed and used in this research is publicly available in the following link: https://github.com/UCA-Datalab/nilm-thresholding.

Declarations

Conflict of interest The authors have no competing interests, or other interests that might be perceived to influence the results and/or discussion reported in this paper.

Ethics approval Not applicable.

References

1. George William H (1992) Nonintrusive appliance load monitoring. In: Proceedings of the IEEE, 80(12):1870–1891, ISSN 15582256. https://doi.org/10.1109/5.192069
2. Christoforos N, Dimitris V (2019) Machine learning approaches for non-intrusive load monitoring: from qualitative to quantitative comparation. Artif Intell Rev 52(1):217–243. https://doi.org/10.1007/s10462-018-9613-7

3. Pedro Paulo Marques do N (2016) Applications of deep learning techniques on NILM. Universidade Federal do Rio de Janeiro, Diss
4. Christoph K and Peter G (2016) Non-intrusive load monitoring: A review and outlook. Lecture Notes in Informatics (LNI). In: Proceedings—Series of the Gesellschaft fur Informatik (GI), 259(1):2199–2210, ISSN 16175468
5. Kolter JZ, Matthew JJ (2011) REDD: a public data set for energy disaggregation research. In: Workshop on data mining applications in sustainability (SIGKDD), San Diego, CA, Citeseer, 59–62
6. Kyle A, Adrian O, Diego B, Derrick C, Anthony R, and Mario B(2012) BLUED: A fully labeled public dataset for event-based non-intrusive load monitoring research. In: Proceedings of the 2nd KDD Workshop on Data Mining Applications in Sustainability (SustKDD), pp 1–5
7. Jack K, William K (2015) The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. Sci Data 2:3. https://doi.org/10.1038/sdata.2015.7
8. Christoph K, Andreas R, Lucas P, Stephen M, Mario B, Wilfried E (2019) Electricity consumption data sets: Pitfalls and opportunities. In: BuildSys 2019 - Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation, pp 159–162. https://doi.org/10.1145/3360322.3360867
9. Christoph K, Stephen M, Wilfried E (2020) Towards comparability in non-intrusive load monitoring: on data and performance evaluation. arXiv, ISSN 23318422
10. Jorge RH, Alvaro LM, Alberto B, Daniel de la I, Villarrubia G, Juan CR, Rita C (2018) Non Intrusive Load Monitoring (NILM): a state of the art. In: International Conference on Practical Applications of Agents and Multi-Agent Systems, pp 125–138, ISBN 978-3-319-61577-6. https://doi.org/10.1007/978-3-319-61578-3_12
11. Anthony F, Nerey HM, Shubi K, Kisangiri M (2017) A survey on non-intrusive load monitoring methodies and techniques for energy disaggregation problem. arXiv, ISSN 23318422
12. Kim H, Marwah M, Arlitt M, Lyon G, Han J (2011) Unsupervised disaggregation of low frequency power measurements. Proc SIAM Conf Data Min 11:747–758. https://doi.org/10.1137/1.9781611972818.64
13. Zico Kolter J, Tommi J (2012) Approximate inference in additive factorial HMMs with application to energy disaggregation. J Mach Learn Res 22:1472–1482
14. Ruoxi J, Yang G, and Costas JS (2016) A fully unsupervised non-intrusive load monitoring framework. In: 2015 IEEE International Conference on Smart Grid Communications, SmartGridComm 2015, pp 872–878. https://doi.org/10.1109/SmartGridComm.2015.7436411
15. Himeur Y, Alsalemi A, Bensaali F, Amira A, Al-Kababji A (2022) Recent trends of smart nonintrusive load monitoring in buildings: A review, open challenges, and future directions. Int J Intell Syst 37(10):7124–7179
16. Kim J, Le Thi-Thu-Huong TH, Kim H (2017) Based nonintrusive load monitoring, on advanced deep learning and novel signature. Comput Intell Neurosci. https://doi.org/10.1155/2017/4216281
17. Odysseas K, Christoforos N, Dimitris V (2018) Sliding window approach for online energy disaggregation using artificial neural networks. In: ACM International Conference Proceeding Series, vol 1, pp 1–6, ISBN 9781450364331. https://doi.org/10.1145/3200947.3201011
18. Maria K, Nikolaos D, Anastasios D, Athanasios V, Eftychios P (2019) Bayesian-optimized bidirectional LSTM regression model for non-intrusive load monitoring. In: ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2019-May(March 2020):2747–2751, ISSN 15206149. https://doi.org/10.1109/ICASSP.2019.8683110
19. Kalthoum Z, Mohamed Lassaad A, Ridha B (2022) Lstm-based reinforcement q learning model for non intrusive load monitoring. In: Advanced Information Networking and Applications: Proceedings of the 36th International Conference on Advanced Information Networking and Applications (AINA-2022), Vol 3, pp 1–13. Springer
20. Lamprini K, Christoforos N, Dimitris V (2019) Imaging time-series for NILM. Commun Comput Inf Sci 1000:188–196. https://doi.org/10.1007/978-3-030-20257-6_16
21. Luca M, Marino M, Simone M (2020) Non-intrusive load disaggregation by convolutional neural network and multilabel classification. Appl Sci (Switzerland) 10(4):17. https://doi.org/10.3390/app10041454
22. Huan C, Yue-Hsien W, Chun-Hung F (2021) A convolutional autoencoder-based approach with batch normalization for energy disaggregation. J Supercomput 77:2961
23. Mohamed A, Jérémie D, Rim K, Jesse R (2023) Conv-nilm-net, a causal and multi-appliance model for energy source separation. In: Machine Learning and Principles and Practice of

Knowledge Discovery in Databases: International Workshops of ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part II, Springer, pp 207–222
24. Ducange P, Marcelloni F, Antonelli M (2014) A novel approach based on finite-state machines with fuzzy transitions for nonintrusive home appliance monitoring. IEEE T Ind Inf 10(2):1185–1197
25. Kong W, Dong ZY, Hill DJ, Luo F, Xu Y (2016) Improving nonintrusive load monitoring efficiency via a hybrid programing method. IEEE Trans Ind Inf 12(6):2148–2157
26. Chuan Choong Y, Chit Siang S, Vooi Voon Y (2015) A systematic approach to on-off event detection and clustering analysis of non-intrusive appliance load monitoring. Front Energy 9(2):231–237
27. Arfa Y, Shoab Ahmed K (2018) Unsupervised event detection and on-off pairing approach applied to nilm. In: 2018 international conference on frontiers of information technology (FIT), pp 123–128. IEEE
28. Qi L, Kondwani Michael K, Xiaodong L, Mingxu S, Nigel L (2019) Low-complexity non-intrusive load monitoring using unsupervised learning and generalized appliance models. IEEE Trans Consum Electron 65(1):28–37. https://doi.org/10.1109/TCE.2019.2891160
29. Jie L, Liu S, Cai L, Xiong C, Tu G (2023). A multi-task learning model for non-intrusive load monitoring based on discrete wavelet transform. J Supercomput. https://doi.org/10.1007/s11227-022-05000-6
30. Daniel P, David G-U (2021) Non-intrusive load monitoring using multi-output cnns. In: 2021 IEEE Madrid PowerTech, pp 1–6
31. Xiyue W, Wei L (2022) Mtfed-nilm: Multi-task federated learning for non-intrusive load monitoring. In: 2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), pp 1–8
32. Lucas P, Nuno N (2018) Performance evaluation in non-intrusive load monitoring: Datasets, metrics, and tools-A review. Wiley Interdiscip Rev Data Min Knowl Discov 8(6):1–17. https://doi.org/10.1002/widm.1265
33. Stephen M, Fred P (2015) Nonintrusive load monitoring (NILM) performance evaluationa: a unified approach for accuracy reporting. Energy Effic 8(4):809–814. https://doi.org/10.1007/s12053-014-9306-2
34. Antonio R, Alvaro H, Jesus Jesús U, Maria R, Juan G, Juan G (2019) NILM techniques for intelligent home energy management and ambient assisted living: A review. Energies 12(11):1–29. https://doi.org/10.3390/en12112203
35. James M et al (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Oakland, pp 281–297
36. Sanket D, Rabei A, Abdun M, Naveen C, Seungmin R (2019) Multi-state energy classifier to evaluate the performance of the NILM algorithm. Sensors (Switzerland) 19(23):1–17. https://doi.org/10.3390/s19235236
37. Jack K, William K (2015b) Neural NILM. In: BuildSys 2015 - Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built, pp 55–64. https://doi.org/10.1145/2821650.2821672
38. Deyvison de PP, Adriana Rosa GC, Deyvison de PP, Adriana RGC (2017) Convolutional neural network applied to the identification of residential equipment in nonintrusive load monitoring systems. In: 3rd International Conference on Artificial Intelligence and Applications, pp 11–21 https://doi.org/10.5121/csit.2017.71802
39. Yang Y, Zhong J, Li W, Aaron Gulliver T, Li S (2020) Semisupervised multilabel deep learning based nonintrusive load monitoring in smart grids. IEEE Trans Ind Inf 16(11):6892–6902
40. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S (2014) Dragomir Anguelov. Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions, Dumitru Erhan
41. Changho S, Sunghwan J, Jaeryun Y, Hyoseop L, Taesup M, Wonjong R (2019) Subtask gated networks for non-intrusive load monitoring. In: 33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, 33:1150–1157, ISSN 2159-5399. https://doi.org/10.1609/aaai.v33i01.33011150
42. Oliver P, Grant F, April H, Nipun B, Jack K, Amarjeet S, William K, Alex R (2016) Dataport and NILMTK: A building data set designed for non-intrusive load monitoring. 2015 IEEE Global

Conference on Signal and Information Processing, GlobalSIP 2015, pages 210–214. https://doi.org/10.1109/GlobalSIP.2015.7418187

43. Wilhelm K, Christian B, Silvia S (2015) Household Occupancy Monitoring Using Electricity Meters. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2015), Osaka, Japan