Energy and AI xxx (xxxx) xxx

Contents lists available at ScienceDirect

Energy and AI

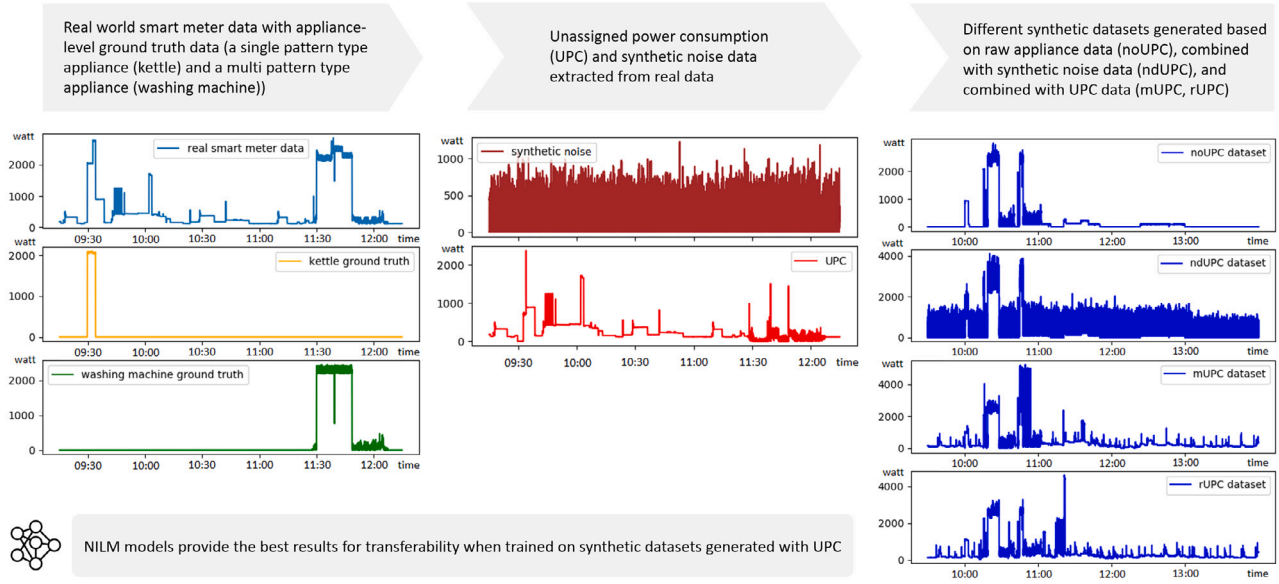
journal homepage: www.elsevier.com/locate/egyai

Generation of meaningful synthetic sensor data — Evaluated with a reliable transferability methodology

Michael Meiser *, Benjamin Duppe, Ingo Zinnikus

Deutsches Forschungszentrum für künstliche Intelligenz (DFKI), Stuhlsatzenhausweg 3, Saarbrücken, 66123, Saarland, Germany

GRAPHICAL ABSTRACT



HIGHLIGHTS

- Generation of multiple meaningful synthetic energy datasets using different concepts.
- The inclusion of UPC is crucial for generating meaningful synthetic energy data.
- Provision of a reliable evaluation methodology for comparing the quality of energy data.
- The methodology developed provides a novel metric based on transferable NILM models.
- NILM models trained on UPC enhanced synthetic data outperform those trained on real data.

ARTICLE INFO

Keywords:
Smart home
Synthetic sensor data
Energy data

ABSTRACT

As households are equipped with smart meters, supervised Machine Learning (ML) models and especially Non-Intrusive Load Monitoring (NILM) disaggregation algorithms are becoming increasingly important. To be robust, these models require a large amount of data, which is difficult to collect. Consequently, the generation

* Corresponding author.
E-mail addresses: michael.meiser@dfki.de (M. Meiser), benjamin.duppe@dfki.de (B. Duppe), ingo.zinnikus@dfki.de (I. Zinnikus).

https://doi.org/10.1016/j.egyai.2023.100308
Received 25 July 2023; Received in revised form 19 September 2023; Accepted 11 October 2023
Available online 13 October 2023

Transfer learning
Evaluation methodology
Machine learning
Neural networks
NILM
Seq2point
WindowGRU
DAE
Seq2seq
RNN

of meaningful synthetic data is becoming more relevant. We use a simulation framework to generate multiple datasets using different techniques and evaluate their quality statistically by measuring the performance of NILM models for transferability. We demonstrate that the method of data generation is crucial to train ML models in a meaningful way. The experiments conducted reveal that adding noise to the synthetic smart meter data is essential to train robust NILM models for transferability. The best results are obtained when this noise is derived from unknown appliances for which no ground truth data is available. Since we observed that NILM models can provide unstable results, we develop a reliable evaluation methodology, based on Cochran's sample size. Finally, we compare the quality of the generated synthetic data with real data and observe that multiple NILM models trained on synthetic data perform significantly better than those trained on real data.

Abbreviations

AI	Artificial Intelligence
ML	Machine Learning
NILM	Non-Intrusive Load Monitoring
SynTiSeD	Synthetic Time Series Data Generator
UPC	Unassigned Power Consumption
NILMTK	Non-Intrusive Load Monitoring Toolkit
PCP	Power Consumption Pattern
MAE	Mean Absolute Error
EUH	Error on Unseen Houses
MAEUH	Mean Absolute Error on Unseen Houses
SD	Standard Deviation
S2P	Sequence to Point
S2S	Sequence to Sequence
DAE	Denoising Autoencoder
RNN	Recurrent Neural Network
wGRU	Window Gated Recurrent Units
M-W-U	Mann–Whitney-U test

1. Introduction

With global climate change and the associated increase in energy costs, awareness of energy and energy efficiency is growing rapidly in both the industrial and private sectors [1–4]. This is accompanied by a substantial growth in demand for methods and applications that help monitor and optimize power consumption, with many solutions already using *Artificial Intelligence* (AI) concepts [5,6]. These AI methods include *Machine Learning* (ML) models such as energy prediction algorithms [7], human activity recognition models [8] or *Non-Intrusive Load Monitoring* (NILM) disaggregation algorithms. NILM approaches are able to determine when a particular appliance has consumed energy based on the total consumption of a household [9] and even to recommend better practices for energy usage in specific consumer installations [10]. NILM is often a first step toward more efficient energy management [11], which is expected to reduce overall power consumption [12].

Recently, households are more and more being equipped with a *smart meter*, an intelligent electricity meter that measures the total electricity consumption, as they are even becoming mandatory in some countries, such as Germany, where the law requires their widespread use in households and companies by 2032 [13]. Smart meters are expected to provide benefits to consumers, such as understanding their energy usage [14] and reducing costs through more accurate and immediate feedback on energy usage [15].

ML models in general, and thus NILM disaggregation algorithms, require a large amount of data to become robust and functional, but in practice this data is often not available to a sufficient extent [16], slowing down the speed of their innovation. The collection and preparation of aggregated energy data from real households (smart meter data) is complex, expensive, requires a lot of time and is even often not possible due to data privacy issues. Moreover, the ground truth data of the individual appliances that compose the readings of the smart meters is not always complete or even available, and collecting this data is also problematic. In practice, it is necessary to use already trained models and transfer them to other households, as data acquisition for individual households would be extremely time consuming and expensive, making it effectively impossible to train individual models per household.

For this reason, transferability is vastly more challenging for NILM tasks than it is in other domains. Training with relatively few households is expected to result in models whose performance is not robust to transferability. However, for the reasons mentioned above, it is practically impossible to collect data from a sufficient number of households to train robust and general models.

An obvious approach for generating more labeled smart meter data is to use a simulation. Simulations offer many advantages, such as the ability to generate theoretically infinite amounts of annotated data, or to generate data from critical situations that would be too dangerous to collect in reality. Consequently, they are used more and more in almost all application domains, such as heating energy of buildings [17–19] or Industry 4.0 [20,21]. Using a simulation for generating smart meter data allows to circumvent problems such as missing ground truth data, different sampling rates, holes in the data, limited labeling, and significant differences in available data [16] that occur in collected real-world datasets. Therefore, we developed the *Synthetic Time Series Data Generator* (SynTiSeD) [22], a framework that allows generating synthetic data based on real data.

However, simulations also face challenges that need to be addressed. They allow the generation of large amounts of data, but it is also necessary to ensure that this data is meaningful and adequately reflects a domain. This raises the following questions, which form the focus of this thesis:

- How to generate synthetic data that allows to meaningfully train transferable NILM models?
- How to reliably evaluate the quality of synthetic data using NILM models?

In order to generate useful data, it is necessary to analyze which factors affect the NILM models and how. We demonstrate that it is essential to understand how the synthetic data must be generated to train transferable ML models. In this work, we highlight the impact of unknown appliances in the smart meter data, for which we introduce the concept of *unassigned power consumption* (UPC).[1] UPC pertains to appliances for which no ground truth data is available. As we will demonstrate in this work, the UPC has a significant impact on how NILM models can be trained in a reliable manner.

Ultimately, a major goal in research on appliance disaggregation is to improve the performance of NILM models. There are at least two common ways to improve NILM models: by optimizing the model architecture [23–29] and, on the other hand, given the sparse data available as explained above, by improving the data. These approaches are complementary to each other. Also complementary to each other

[1] UPC acts as noise for the NILM models, i.e. recorded power consumption that is not part of the appliance that the model should detect.

is the approach to validation and evaluation of the two approaches: when NILM models are compared, one or several fixed, stable datasets are used for evaluation, ideally a benchmark to ensure availability and reproducibility. Conversely, and this is implemented in the present work, different datasets are evaluated for quality by submitting them to the same model (or models) for training. Fundamental in both cases, of course, is that the measuring rod itself remains stable during the measurements. As it turns out, in the case of comparing datasets, there is the problem that ML models differ due to the stochastic nature of the training process. Not having taken this into account, or at least having stressed it, is the shortcoming of the approaches available, which we avoid in this paper with a methodology to reliably compare NILM algorithms. We use Cochran's sample size measure to ensure that there is a sufficient number of trained models to obtain reliable results. To ensure reproducibility, we furthermore use the widely used framework NILMTK for the creation of NILM models, which enjoys a high reputation in the NILM research community.

We demonstrate that the analyzed NILM models, trained on synthetic data generated by our methods and *SynTiSeD*, provide better results in terms of transferability than models trained on real data. Anticipating the results, we even show that the synthetic data partially yields very highly significant improvements.

The key contributions of this research are as follows:

- We demonstrate that UPC is crucial for generating meaningful synthetic energy data that can be used to train robust NILM models for transferability.
- We develop a comprehensive and reliable evaluation methodology, not previously applied in this field, to confidently compare the results of NILM models.
- We generate meaningful synthetic energy datasets and evaluate their quality against each other and against real-world data using our reliable evaluation methodology.
- We present evidence that NILM models trained on synthetic data can significantly outperform models trained on comparable real-world data, proving that synthetic data is suitable for training robust NILM models.

This work is organized as follows. It starts with an analysis of related work (Section 2). We then explain in detail how smart meter data is structured, the concept of UPC, and how we are able to generate synthetic data (Section 3), before explaining the methodology we developed to reliably compare the quality of energy data using NILM algorithms (Section 4). We then describe how we prepare the data for our synthetic datasets and how we design the experiments to evaluate the impact of unknown appliances in smart meter data on training robust ML models (Section 5). Finally, we present the results of the extensive evaluation using statistical methods as well as the developed methodology (Section 6) and discuss the results of our work (Section 7).

2. Related work

In practice, the use of effective and robust *Non-Intrusive Load Monitoring* (NILM) disaggregation algorithms provides many benefits, hence they have become an essential part of smart home research and even industrial research [30]. NILMTK (Non-Intrusive Load Monitoring Toolkit) with the latest update NILMTK-contrib [31] is a freely available github project,[2] and provides the ability to work with many of these algorithms. Because of its free accessibility, which ensures comparability, many authors in the community use NILMTK to disaggregate energy data [30,32–34].

Most NILM studies are performed only for the same data domain, but recently the importance of transfer learning has become popular [33–38]. There is work investigating transfer learning of appliances, examining whether the patterns learned by one appliance can be transferred to other appliances [35]. Most of the papers investigate whether NILM algorithms of an appliance can be transferred to different domains (e.g., other households), a topic we understand as cross-domain transferability. In this work, we focus on cross-domain transferability, hereafter abbreviated as transferability. We understand transferability as the ability of a model developed for a specific dataset to be generalized to new households not present in the training set [39].

Tools and frameworks already exist to generate synthetic power consumption data, such as the *Automated Model Builder for Appliance Loads* (AMBAL) [40]. In this tool, individual consumption patterns are sequenced a sufficient number of times to optimize the resulting segments with a mean absolute percentage error. This can cause the individual segments to become very small, resulting in very similar patterns. The produced synthetic data is evaluated using the Combinatorial Optimization algorithm of NILMTK, which is implemented as a mathematical minimization problem. The authors do not describe whether they use UPC or a similar concept to generate their data.

SmartSim [41] creates power consumption traces for appliances by combining an appliance model that captures the pattern of power consumption in the active state, with a usage model that specifies the frequency, duration, and timing of the activity.[3] By combining data from individual appliances, the framework then generates aggregated data for a simulated house. However, the techniques used require a great deal of manual effort. Although the authors evaluate their resulting data, they only do so using Factorial Hidden Markov Models and Combinatorial Optimization NILM algorithms, which are no longer state of the art. The authors also indicate that the results of the NILM algorithms for the five most used appliances, trained on their generated data, are not as high as the models trained on comparative data.

With ANTgen, the authors introduced a tool (available on github[4]) that was used to create synthetic data based on real-world data [32]. To evaluate the generated synthetic data, different NILM models were trained and tested on this data. Also, for comparison, different NILM models were trained and tested exclusively on the real data (on freely available datasets). However, the authors only examine whether NILM models can be trained on the generated data, they do not consider the transferability of NILM models.

Kelly and Knottenbelt [33] generate synthetic data by copying real-world power consumption patterns (PCPs) and combining them, without considering at what time an appliance is typically turned on or whether appliances are typically turned on in sequence. The authors note that a more realistic simulator accounting for the temporal structure of appliance activation could improve the performance of deep neural networks compared to their naive approach. The NILM models examined are trained on both synthetic and real aggregated data in a 50:50 ratio, while real data is used for validation and testing. Their work also tests the transferability of their models. To do this, they use unknown households for testing, for which they use NILMTK. The authors notice that training with a mixture of synthetic and real aggregate data seems to improve the ability of the network to generalize to unknown households.

SMACH is a multi-agent based simulator (not publicly available) that generates synthetic action sequences for each time sample, consisting of a list of all appliances in a household and an indicator of whether these appliances consume energy [42]. Together with real PCPs collected from their own data (27 households, not freely published), these sequences are used to generate synthetic data. The authors demonstrate that synthetic data improves the performance of NILM in transfer learning for a seq2seq and a seq2point model. However, they do not describe how exactly their data is constructed, e.g. whether they use power consumption data of unknown appliances. Apparently, only appliances with ground truth data are used.

[2] NILMTK-contrib: https://github.com/nilmtk/nilmtk-contrib.

[3] SmartSim: https://github.com/klemenjak/smartsim.

[4] ANTgen: https://github.com/areinhardt/antgen.

Some work exists where synthetic datasets are created and then assessed in comparison to real data using stochastic metrics [43]. However, in our opinion, purely statistical methods for comparing datasets are not sufficient to prove that NILM models can be meaningfully trained, because data may be similar according to a given stochastic metric, but do not necessarily lead to comparable NILM results when trained on it. Unless a clear-cut correspondence between a selected stochastic metric for data and the performance of NILM models on this data is established, this approach is not reliable. Klemenjak et al. [39] present metrics to measure the transferability of NILM models, but we consider them also insufficient due to the instability of the models given their stochastic nature, which makes it practically impossible to reliably compare individual NILM algorithms (see Section 4).

None of the previously listed works [33–35,39,43], nor any other papers investigating transferability in the context of NILM [36–38,44], describe a methodology for how to evaluate data using NILM, taking into account their stochastic nature. They do not describe whether they repeated their experiments, i.e., whether they trained, validated, and tested models on the same data with the identical parameters multiple times to eliminate measurement tolerances, or their NILM disaggregation models used might generate different outcomes. Furthermore, none of the papers cited above mention whether they use UPC or some equivalent to generate their data or indicate the importance of UPC for the transferability of NILM models. Apart from the papers referenced in this section, we are not aware of any work that addresses the described issues in the NILM context.

3. Synthetic energy data

The following section describes in more detail the approach we use to generate the synthetic energy datasets to highlight the impact of unknown appliances in smart meter data. First, we categorize the appliances that contribute to the aggregated smart meter data of a typical household (see Section 3.1). We then introduce the concept of *unassigned power consumption* (see Section 3.2), before describing *SynTiSeD*, the simulation framework we use to generate the synthetic data for this work (see Section 3.3).

3.1. Appliance categorization

In order to generate meaningful synthetic energy data, it is important to understand the components that contribute to a household's aggregated power consumption and which classes of appliances are included. One approach is to categorize appliances according to their *power consumption patterns* (PCPs). [43] define a total of four categories, which are explained in more detail below.

- ***Constantly-On Patterns*** include appliances, such as a router, that consume energy constantly and without any downtime.
- ***Periodical Patterns*** include appliances, such as a refrigerator, a freezer or a heating pump, which work independently and have periodical PCPs.
- ***Single Patterns*** include appliances that require a user to either operate or start a specific program. This means that a user activates an appliance that performs a specific task and turns it off, or the appliance is turned off when the task is completed. This type of appliance includes those that have a similar PCP for each operation, such as an electric kettle or a lamp. The main characteristics (e.g. peak consumption and shape) of the pattern can be predicted fairly well, although the length of the pattern can vary due to external factors.
- ***Multi Patterns*** include appliances, such as a washing machine or a dishwasher, with different operations and therefore different PCPs, which can vary not only in length but also in process steps. For example, a washing machine is capable of running different programs, e.g. a short and a long wash program, which can result in completely different PCPs.

Table 1
Power consumption pattern types with example appliances.

Pattern type	Appliances
Constantly-On	Router, home assistant
Periodical	Refrigerator, freezer, heating pump
Single	Electric kettle, hair dryer, lamp, monitor, electric oven, pocket radio
Multi	Washing machine, dishwasher, microwave iron, coffee machine, fan, TV

Table 1 summarizes the four different appliance pattern types, each with a number of example appliances that can be found in a typical household. We expect that a framework that is able to generate synthetic power consumption data should cover all four appliance pattern types presented.

3.2. Unassigned power consumption

In practice, publicly available datasets do not include ground truth data from all the appliances that contribute to the total power consumption of a household (see the public energy datasets GeLaP [45], REFIT [46], Eco [47], GREEND [48], ENERTALK [49], IEDL [50], iAWE [51] or UK-DALE [52]). There can be many reasons for this, such as cost or the fact that not all appliances are well suited for recording. However, this means that there are always appliances in a household that contribute to the smart meter data, but there is no ground truth data available.

When generating synthetic data, using only a few known appliance-level data can result in very simple smart meter data without the typical noise. During our work, we observed that training on clean data does not contribute to better performance of NILM models in terms of transferability (see Section 6). To address this issue, we introduce the concept of *unassigned power consumption* (UPC).

We extract the UPC of a respective household from the unknown appliances (the appliances for which no ground truth data is available). We do this by subtracting the ground truth data of all known appliances of a household from its smart meter data on a daily basis.

As a result, the total active power consumption of a household $pc(t)$ at a given timestamp t is equal to the sum of the power consumption of all N_{ka} known appliances $ap_i(t)$ plus a UPC $upc(t)$ (see Eq. (1)).

$$pc(t) = \sum_{i=1}^{N_{ka}} ap_i(t) + upc(t) \tag{1}$$

With our approach of extracting the UPC, unsynchronized time series, i.e. time shifts between the ground truth data and the smart meter data, can occur due to the different recording sensors, resulting in negative values. Since negative values are not realistic in smart meter data, all negative values are set to zero.

An example of how UPC is generated is illustrated in Fig. 1. It demonstrates a sample section of real smart meter data from the GeLaP dataset, the related ground truth data of a single pattern type appliance (kettle) and a multi pattern type appliance (washing machine), and the extracted UPC. In this example, the kettle and the washing machine are the only appliances of a household that are known, i.e. for which ground truth data is available. We generate the UPC by subtracting the ground truth data of the known appliances, i.e. the kettle and the washing machine, from the smart meter data of the household and set possible resulting negative values to zero.

3.3. SynTiSeD

The *SynTiSeD* framework used for this paper is a multi agent simulation tool [22,53]. The latest version of *SynTiSeD* is described in more

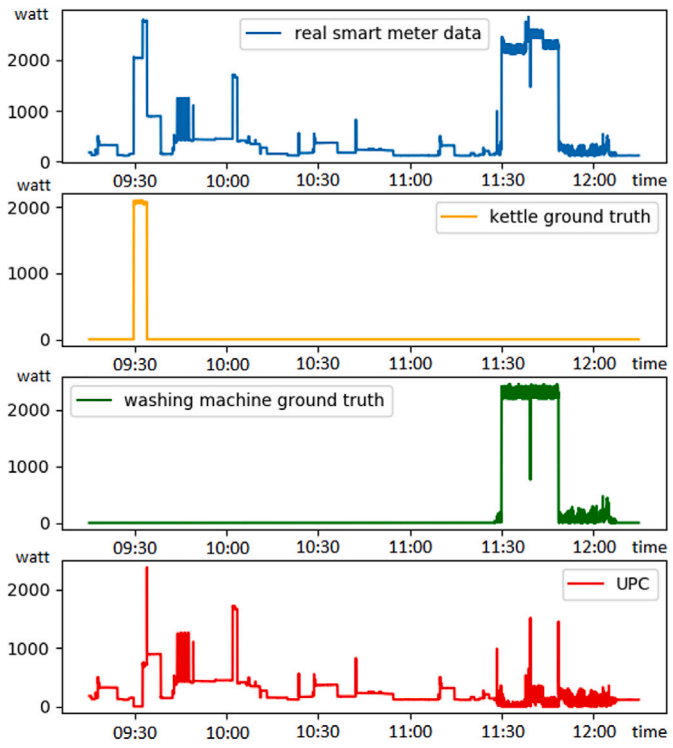


Fig. 1. Comparison of an example section of real smart meter data of the GeLaP dataset, the ground truth data of a single pattern type appliance (kettle) and a multi pattern type appliance (washing machine) and the UPC extracted from them.

detail on github.[5] *SynTiSeD* is able to create meaningful synthetic smart meter data, which in turn can be used for training ML models [22].

SynTiSeD allows to create a number of *households* ($H = h_1, h_2, \dots, h_j$), to each of which different *residents* ($R = r_1, r_2, \dots, r_k$) can be added and to which *appliances* ($D = d_1, d_2, \dots, d_l$) can be attached. A resident is an entity that acts and interacts with the appliances in a household using *action sequences*, where each sequence consists of *actions* ($A = a_1, a_2, \dots, a_m$).

During a simulation run $y \in Y$, a resident agent $r \in R$ changes its state step by step with an iterative execution of actions $a \in A$, where a state is one of the appliances $d \in D$ contained therein [54]:

$$y \in Y : d_a \xrightarrow{a_1 \in A} d_b \xrightarrow{a_2 \in A} d_c \xrightarrow{a_3 \in A} .. \tag{2}$$

SynTiSeD generates synthetic data by executing the sequences of actions assigned to each resident in each household, and thus interacting with the associated virtual appliances. To do this, *SynTiSeD* aggregates the data from each corresponding timestamp of all permanent virtual appliances connected to the household to the virtual smart meter. Permanent appliances include all appliances of the *constantly on* and *periodical* appliance pattern types (see Section 3.1) that typically consume energy without requiring direct human activation. At the same time, *SynTiSeD* iterates over each timestamp of the day specified by the sampling rate of the simulation, constantly checking whether the residents of the household trigger an appliance (human-induced virtual appliance, containing *single* and *multi* appliance pattern types) over the iterated timestamp; and if an appliance is triggered, *SynTiSeD* pulls the relevant stored PCP data and merges it into the virtual smart meter. Thus, one day of the virtual smart meter consists of the data of all permanent virtual appliances, as well as the data of all human-induced virtual appliances used by all residents in the household on that day.

[5] *SynTiSeD*: https://github.com/mmeism/SynTiSeD_research.

For each virtual household *SynTiSeD* returns a series of total power consumption *hpc* at timestamp t over all generated timestamps N_t:

$$hpc(t_1), hpc(t_2), \dots, hpc(t_n), \dots, hpc(t_{N_t}) \tag{3}$$

where each total power consumption *hpc* at timestamp t equals the sum N_{ha} of all human induced appliances *ha* plus the sum N_{pa} of all permanent appliances *pa* plus the UPC (see Eq. (4)).

$$hpc(t) = \sum_{o=1}^{N_{ha}} ha_o(t) + \sum_{q=1}^{N_{pa}} pa_q(t) + upc(t) \tag{4}$$

The sequenced PCPs of each appliance are used together with the extracted action sequences and the UPC to generate synthetic datasets with *SynTiSeD*. Besides the virtual smart meter, for every virtual household *SynTiSeD* simultaneously generates an individual energy history for each appliance used, providing the appliance ground truth. The *SynTiSeD* framework is capable of covering all four appliance pattern types presented in Section 3.1. In addition, *SynTiSeD* is the only framework we are aware of that takes UPC into account for generating synthetic energy data.

4. Evaluation methodology

During our research, we observed that NILM models do not produce stable results when it comes to transferability, even when trained with identical hyperparameters on identical data. Since we noticed that other works do not address the issue of instability of NILM models, especially in terms of transferability (see Section 2), we introduce a new evaluation methodology that takes this fact into account. In this section, we first examine the NILM approach more closely. We then describe our evaluation methodology in more detail (see Section 4.2), before demonstrating the necessity of the methodology with a proof of concept (see Section 4.3).

4.1. Non intrusive load monitoring

NILM approaches are mainly used to determine when and how much a particular appliance consumed energy based on the smart meter data of a household. Considering Eq. (1), the goal of NILM is to most accurately estimate the power consumption of each appliance in a household $F(pc(t))$. F is an operator that generates a sequence of power consumption estimates at timestamp t for each (known) appliance ap_i at timestamp t (see Eq. (5)) [55].

$$F(pc(t)) = ap_1(t), ap_2(t), \dots, ap_i(t), \dots, ap_{N_{ka}}(t) \tag{5}$$

As mentioned in Section 3.2, in practice the sum of all known appliances $ap_i(t)$ does not represent the complete set of household appliances, since existing publicly available datasets do not provide ground truth data for all appliances. To address this issue, we introduced the concept of UPC. If both the smart meter data of a household and the ground truth data of all the individual appliances are available at the same time, the approximation of $F(pc(t))$ can be considered as a supervised learning problem. The estimated power consumption of each known appliance $\hat{ap}_i$ at timestamp t can be calculated using the average power consumption p_i of an appliance i and an estimation of the activation state $\hat{as}_i(t)$ of that appliance at timestamp t (see Eqs. (6) and (7)) [55].

$$\hat{ap}_i(t) = p_i \hat{as}_i(t), \tag{6}$$

where:

$$\hat{as}_i(t) = \begin{cases} 1 & \text{if appliance } i \text{ is in use at timestamp } t \\ 0 & \text{if appl. } i \text{ is not in use at timestamp } t \end{cases} \tag{7}$$

In this work, we focus on using NILM models to disaggregate the power consumption of a household to individual appliances (see Eq. (6)) in order to measure the quality of the training data of the models.

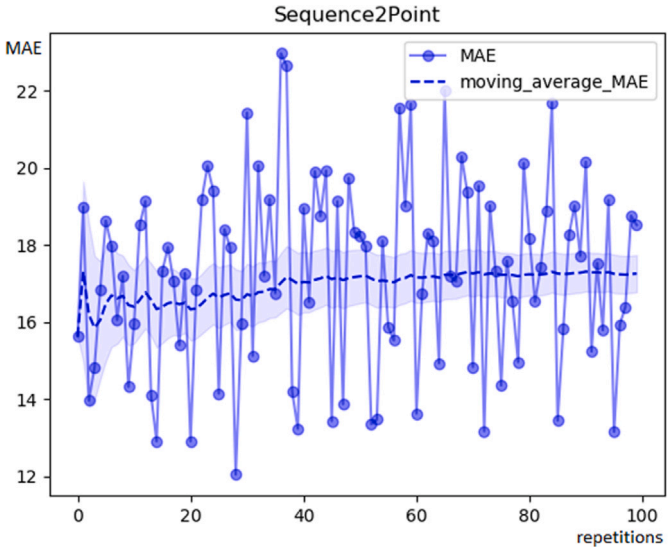


Fig. 2. MAE, Moving Average MAE and confidence interval for the experiment results for a single pattern appliance type (kettle) of 100 seq2point NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

4.2. Methodology

From the point of view of applicability, it is important that ML models for NILM are transferable. Hence, in practice, it is indispensable to use models that are already trained, or at least pretrained, in order to avoid long data acquisition phases when deploying them in a new household.

Many papers evaluate transferability by measuring the performance of one model trained on households different from the test households and comparing it to a model trained and tested on another household [33–38]. None of these papers address the issue of instability of NILM models, especially in terms of transferability (see Section 2). From our point of view, this seems problematic because of the stochastic properties of neural networks. It is a well-known fact that models trained on identical data can perform differently. According to our findings, this is a problem that must be taken into account when evaluating models for NILM transferability.

The performance of NILMTK models (seq2point, seq2seq, DAE, RNN and windowGRU, see Section 5.3) trained on the same data varies substantially in our experiments. Since in NILMTK the tested disaggregation algorithms were implemented as a regression problem, we use the *mean absolute error* (MAE) as comparison metric, which is a commonly used and accepted metric (see [25,32,34,35,39,44,56–59]).

Eq. (8) denotes the MAE, where x_t is the ground truth and $\hat{x}_t$ is the prediction of an appliance at time t.

$$\text{MAE} = \frac{1}{T}\sum_{t=1}^{T}|\hat{x}_t - x_t| \tag{8}$$

Fig. 2 illustrates the MAE, the Moving Average MAE and the confidence interval for 100 results of a NILM model with the same training and test households and the identical hyperparameters. Examples of the results of other NILM models can be found in Appendix A.

The Moving Average MAE represents a series of average MAEs (see Eq. (9)), which itself is equal to the sum of the MAE over all performed experiments divided by the number of performed experiments N_e.

$$\text{Average MAE} = \frac{1}{N_e}\sum_{e=1}^{N_e}\text{MAE}_e \tag{9}$$

In this work, the confidence interval (CI) (see Eq. (10)) is computed at a 95% level, with $\bar{x}$ representing the sample mean, z the confidence level value, s the sample standard deviation, and n the sample size.

$$\text{CI} = \bar{x} \pm z\frac{s}{\sqrt{n}} \tag{10}$$

In order to make well-founded statements about whether models trained on synthetic data or models trained on real data perform better, it is necessary not only to consult the results of a single experiment, but to repeat the experiment to obtain stable results. Of course, this is only feasible to a certain extent, since each experiment requires a corresponding amount of time and computational resources, which is why we need a sample size metric.

To determine the number of necessary experiment repetitions, we use the concept of sample size described by Cochran [60], which is frequently used in empirical studies. The way to calculate the required sample size n_0 is formalized in Eq. (11), where Z is the z-value for the confidence level, which can be found in a Z table, e is the margin of error, p is the proportion of the population, and q is $1 - p$.

$$n_0 = \frac{Z^2 pq}{e^2} \tag{11}$$

For our work, we use a confidence level of 95%, which results in a Z of 1.96, e of 10%, and a p of 0.5, which is the commonly used value when the proportion of the population is unknown at the beginning of the experiments. This results in a sample size of at least 97, which is a feasible number of repetitions in our case. Therefore, we run 100 repetitions to be sure of getting usable and reliable results.

Since we want to validate the transferability of NILM models, we need a specialized metric. Therefore, we adjust and extend the transferability regression metric *error on unseen houses* (EUH) by [39], which measures how well an algorithm generalizes to households that are not part of the training data, with the MAE and the sample size concept proposed by Cochran (see Eq. (11)). The resulting *mean absolute error on unseen houses* (MAEUH) metric is calculated as the sum of all unseen households N_{uh} over the sum of all MAE results with Cochran's sample size n_0 divided by the sample size n_0 divided by the number of all unseen households N_{uh} (see Eq. (12)).

$$\text{MAEUH} = \frac{1}{N_{uh}}\sum_{u=1}^{N_{uh}}\left(\frac{1}{n_0}\sum_{e=1}^{n_0}\text{MAE}_e\right)_u \tag{12}$$

After the experiments are completed, we want to check whether the synthetic data is significantly different from the real data. The two-sample t-test (unpaired t-test or independent t-test) checks whether two independent groups are significantly different. However, for the t-test, the results of the repetitions must be normally distributed [61]. Therefore, we tested the normal distribution using the Shapiro–Wilk test [62] and the test showed that not all result groups are normally distributed. For this reason, the t-test is only partially suitable for determining statistical significance. Consequently, we test significance with the Mann–Whitney-U test, a significance test that does not require a normal distribution of the groups [63].

For the significance level, we set an α-value of 0.05, which is a commonly accepted value. Thus, a p-value $> 5\%$ is *not significant*, a p-value of $\leq 5\%$ is *significant*, a p-value of $\leq 1\%$ is *highly significant*, and a p-value of $\leq 0.1\%$ is a *very highly significant* result [64].

4.3. Case study

The following example illustrates the consequences of training NILM models only once and then comparing them. We use the real data of the GeLaP dataset [45], which includes a total of 20 households (hh). We train one model each for a multi pattern type appliance (washing machine) of the seq2point, seq2seq, DAE, RNN and windowGRU algorithms of the NILMTK (see Section 5.3) on the household *hh*01 and test them on 7 households (*hh*03, *hh*05, *hh*06, *hh*09, *hh*13, *hh*15, *hh*19). Then, we train one model each for the same appliance of the seq2point,

Table 2
MAEUH results (watt) of the NILM Models for the 7 test households of the GeLaP data for a multi pattern type appliance (washing machine), **1 experiment**.

Model	Trained on *hh*01	Trained on *hh*01, *hh*04, *hh*14, *hh*20
S2P	24.208	25.571
S2S	23.691	27.582
DAE	42.184	34.778
wGRU	30.322	34.225
RNN	27.478	30.785

Table 3
MAEUH results (watt) of the NILM Models for the 7 test households of the GeLaP data for a multi pattern type appliance (washing machine), **100 experiments**.

Model	Trained on *hh*01	Trained on *hh*01, *hh*04, *hh*14, *hh*20
S2P	25.797	23.823
S2S	25.847	25.784
DAE	44.814	31.975
wGRU	35.126	28.830
RNN	30.022	26.528

Table 4
MAEUH results (watt) of a multi pattern type appliance (washing machine) from each of the five RUNs, where each RUN represents 100 model results.

Model	RUN 1	RUN 2	RUN 3	RUN 4	RUN 5
S2P	25.797	25.676	25.729	25.769	25.886
S2S	25.847	25.779	25.956	25.946	25.889
DAE	44.814	44.704	44.808	44.789	44.920
wGRU	35.126	35.059	35.232	35.025	35.197
RNN	30.022	29.896	29.961	29.784	30.128

Table 5
Cross significance table — p-values for the Mann–Whitney-U test between each of the 5 RUNs of the multi pattern type appliance (washing machine), where each RUN represents 100 model results.

	Dataset	RUN 1	RUN 2	RUN 3	RUN 4	RUN 5
S2P	RUN 1	1	0.614	0.690	0.652	0.209
	RUN 2	0.614	1	0.925	0.931	0.448
	RUN 3	0.690	0.925	1	0.991	0.375
	RUN 4	0.652	0.931	0.991	1	0.378
	RUN 5	0.209	0.448	0.375	0.378	1
S2S	RUN 1	1	0.715	0.982	0.772	0.887
	RUN 2	0.715	1	0.680	0.527	0.813
	RUN 3	0.982	0.680	1	0.818	0.880
	RUN 4	0.772	0.527	0.818	1	0.693
	RUN 5	0.887	0.813	0.880	0.693	1
DAE	RUN 1	1	0.874	0.719	0.735	0.563
	RUN 2	0.874	1	0.856	0.857	0.648
	RUN 3	0.719	0.856	1	0.998	0.781
	RUN 4	0.735	0.857	0.998	1	0.754
	RUN 5	0.563	0.648	0.781	0.754	1
RNN	RUN 1	1	0.945	0.822	0.556	0.993
	RUN 2	0.945	1	0.880	0.587	0.947
	RUN 3	0.822	0.880	1	0.705	0.811
	RUN 4	0.556	0.587	0.705	1	0.534
	RUN 5	0.993	0.947	0.811	0.534	1
wGRU	RUN 1	1	0.896	0.737	0.919	0.886
	RUN 2	0.896	1	0.606	0.953	0.774
	RUN 3	0.737	0.606	1	0.663	0.843
	RUN 4	0.919	0.953	0.663	1	0.806
	RUN 5	0.886	0.774	0.843	0.806	1

seq2seq, DAE, windowGRU, and RNN algorithms of the NILMTK toolkit on the households *hh*01, *hh*04, *hh*14, and *hh*20 and test them again on the same 7 households (*hh*03, *hh*05, *hh*06, *hh*09, *hh*13, *hh*15, *hh*19) of the real data of the GeLaP dataset. In both cases, the hyperparameters of each model (default NILMTK settings) as well as the test data of each model are identical, the only difference being the number of training households. The results for the MAEUH over all test households are summarized in Table 2.

Comparing the results in Table 2, we can clearly observe that the MAEUH results of 4 of the NILM models (s2p, s2s, RNN, and windowGRU) trained only on *hh*01 are substantially lower and therefore better than the models trained on households *hh*01, *hh*04, *hh*14, and *hh*20. Only the results of the DAE models behave in the opposite way.

If we now compare the results in Table 3 (MAEUH results after 100 experiments), we observe that the MAEUH results of all NILM models trained on *hh*01 are higher and hence worse than the models trained on households *hh*01, *hh*04, *hh*14, and *hh*20. According to the Mann–Whitney-U test for the s2p algorithm the improvement is *significant* (p-value: 0.043), for the s2s it is not significant (p-value: 0.107), for the DAE it is *very highly significant* (p-value: $8.84 * 10^{-66}$), for the RNN it is *very highly significant* (p-value: $1.06 * 10^{-9}$), and for the windowGRU it is *very highly significant* (p-value: $1.56 * 10^{-13}$). Furthermore, comparing the results of Tables 2 and 3, we notice that the results have changed substantially for all algorithms and both training datasets when training 100 models compared to training only a single model.

To check the reliability of our methodology, we repeat the 100 experiments 5 times (RUN) for training household *hh*01 and compare the MAEUH results (see Table 4). Using the Mann–Whitney-U test, no significant differences can be identified between the individual RUNs for any of the NILM algorithms examined (see Table 5). This demonstrates that our methodology provides reliable results for comparing the performance of NILM models, in contrast to comparing only individual NILM models. The example clearly highlights that the results of single NILM models are not sufficient to reliably compare their quality due to their stochastic nature.

5. Evaluation design

Having established the evaluation methodology, we are now able to reliably compare data using NILM algorithms. Furthermore, we are capable of generating single and multi pattern appliance type data using *SynTiSeD*. With these elements, we are now in a position to address the goal of this work, which is to determine the impact of unknown appliances in smart meter data on training robust ML models for transferability.

In this section, we describe how to prepare the data of a public dataset in order to be able to generate synthetic data (see Section 5.1), before we describe in detail how to generate the synthetic datasets we used for the evaluation (see Section 5.2). After that, we describe the NILM disaggregation algorithms we selected for the evaluation (see Section 5.3). Finally, we explain how we set up the experimental environment to achieve the goal of this work (see Section 5.4).

5.1. Data preparation

As already described, *SynTiSeD* uses real appliance data to generate virtual smart meter data (see Section 3.3). For this study, the appliance data is derived from the GeLaP dataset [45]. An advantage of this dataset is that the appliances contained therein are available separately (ground truth data) as well as aggregated (smart meter data). There are a total of 20 households in the GeLaP dataset, each containing smart meter data and ground truth data from 10 appliances per household, with these 10 appliances being different in each household. The GeLaP data also provides relatively high resolution (1 Hz), hence it can be used to generate very accurate synthetic data. In addition, the GeLaP dataset is publicly accessible, ensuring comparability with other work. Of course, any other dataset that meets the properties mentioned above can be used.

We focus our NILM evaluation on appliances that require user activation because they represent the vast majority of appliances in a typical household, resulting in two appliance types (see Section 3.1). One is the *single pattern type* representing appliances with a typical power consumption pattern (PCP) with constant main characteristics.

Table 6
GeLaP data used.

Household	Kettle	Washing machine
*hh*01	2020 03/01–03/14	2020 03/01–03/14
*hh*03	–	2019 12/04–12/17
*hh*04	2020 01/04–03/17	2020 01/04–03/17
*hh*05	–	2019 10/01–10/14
*hh*06	–	2019 11/10–11/23
*hh*07	2019 11/01–11/14	–
*hh*08	2020 01/03–01/31	–
*hh*09	–	2019 09/07–09/20
*hh*11	2019 12/12–12/29	–
*hh*13	–	2020 01/04–01/17
*hh*14	2020 04/01–04/14	2020 04/01–04/14
*hh*15	–	2020 02/01–02/14
*hh*17	2020 01/07–01/31	–
*hh*19	–	2020 03/12–03/25
*hh*20	–	2019 12/17–12/30

The other is the *multi pattern type*, which includes appliances with multiple programs and more complicated PCPs. Since an extensive evaluation with all the necessary details for all instances of both appliance types would be beyond the scope of this work, we examined one appliance each that is representative of its pattern type. We select the particular instances kettle and washing machine, where the kettle represents the single pattern type, while the washing machine represents the multi pattern type (see Section 3.1).

The electric kettle and the washing machine are frequently used in the GeLaP dataset, so there is sufficient data available. Out of the 20 households in the GeLaP dataset, 10 of them include a kettle, but 3 of them are not usable because they have too few PCPs to learn from this data, leaving 7 households with 14 days of data each that we can use. Among the 20 households provided in the GeLaP dataset, 14 contain a washing machine, but 3 of them are unusable because there is no or only one PCP, leaving 11 households. From each household, 14 days of data were extracted (see Table 6).

In total, we synthesize five households (*hh*01, *hh*04, *hh*07, *hh*14, *hh*15), so there are four synthetic households that contain an electric kettle and four households that contain a washing machine. In the following experiments, these households will serve as training and validation data. The remaining real households of the GeLaP dataset are not synthesized and were used as test households for the evaluation. For the kettle we use *hh*08, *hh*11 and *hh*17 as test households, for the washing machine we use *hh*03, *hh*05, *hh*06, *hh*09, *hh*13, *hh*19 and *hh*20.

In the data preparation process, the smart meter data as well as the individual appliance data for all households were first resampled to 1 second (mean) and forward filled to close potential data gaps. For each training and validation household, the appliances for which ground truth data exists were sequenced with different parameters using a custom sequencer, so that we get individual PCPs for each appliance's active phase. There are exceptions for a few appliances, such as 'radio' in *hh*04, which do not have clear individual PCPs, but we interpret them as background power consumption. These appliances, including constantly-on and periodical pattern type appliances (see Section 3.1), were sequenced on a day-to-day basis.

During the sequencing process, we also recorded the times at which each appliance was activated. Using these, we create action sequences that allow us to generate the synthetic data.

To calculate the UPC in the respective households of the GeLaP data, we subtract the ground truth data of all 10 known appliances from the smart meter data on a daily basis (see Section 3.2). Any resulting negative values are set to zero.

In addition, we statistically generate *synthetic noise* to provide a comparison to the UPC. For each calculated day of UPC, we generate a normal distribution with the mean and standard deviation of the UPC and use this distribution as *synthetic noise*. Since this type of generation can result in negative values that do not occur in realistic smart meter data, they are set to zero. Fig. 3 illustrates an example of a section of the UPC and the synthetic noise generated from it.

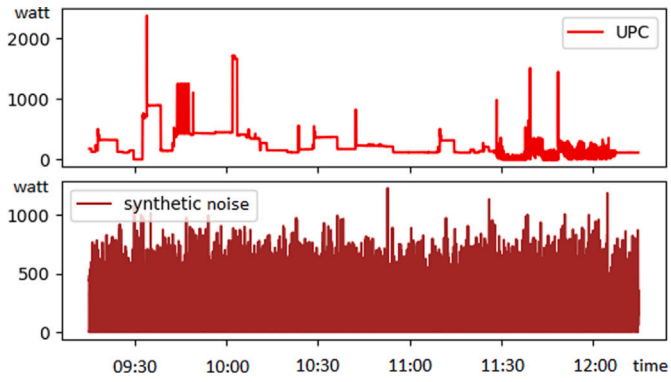


Fig. 3. Comparison of an example section of the extracted UPC and synthetic noise generated from it.

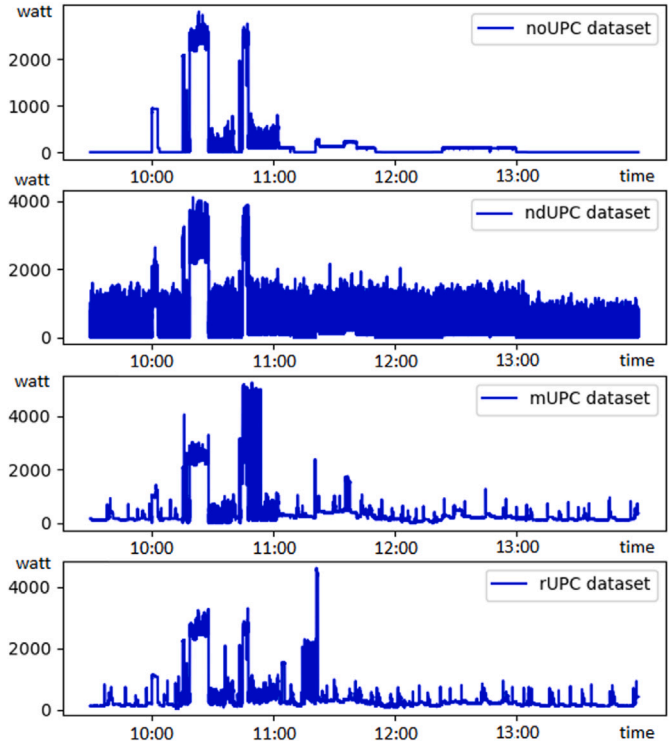


Fig. 4. Comparison of an example section of the four synthetic datasets *noUPC*, *ndUPC*, *mUPC* and *rUPC*.

5.2. Synthetic dataset generation

Since the goal of this work is to make a statement about the effects of the UPC on the transferability of the NILM algorithms, in total four datasets are generated in the following (see Fig. 4). The sequenced PCPs of each appliance are used together with the extracted action sequences, the synthetic noise and the UPC to generate synthetic datasets with *SynTiSeD* (see Section 3.3).

To create the first dataset, hereafter called *noUPC*, for each household we generate the same number of days (two weeks) with the same sampling rate (1 Hz). For each day, we use the real action sequences of the day, i.e. all appliances for which ground truth data is available were activated at the same time as in the real data. At the different times when the actions occur, the sequenced PCPs are then randomly selected out of the PCP database. We use every PCP from the real data to ensure that every PCP occurs in the synthetic data. This dataset was generated as a baseline and does not include any form of UPC.

The second dataset, hereafter called *ndUPC* (normal distributed mapped noise), is based on the *noUPC* dataset. It extends this dataset by assigning the generated synthetic noise of a specific day to the action sequence of the same day. This dataset will be used to determine whether ML models become more robust in terms of transferability when trained and validated on synthetic data containing some type of noise. The hypothesis is that the performance of NILM algorithms will be improved if the smart meter data contains some form of background noise so that the power consumption patterns of individual appliances cannot be clearly identified by the algorithm.

The third dataset, hereafter called *mUPC* (mapped UPC), is also based on the *noUPC* dataset and is basically created like the *ndUPC* dataset, except that a different UPC is used. Instead of using the synthetic noise, this dataset is created with UPC where all negative values are set to zero (see Section 3.2). The purpose of this dataset is to test whether it matters for the transferability of NILM models how exactly the UPC is constructed and whether it matters if the UPC is close to the real data UPC.

The fourth dataset, hereafter called *rUPC* (random UPC), is again based on the *noUPC* dataset. In contrast to the *mUPC* dataset, here the UPC of a specific day is not added to the action sequence of the same day. Instead, the UPC is added randomly. Again, we use the UPC with the negative values set to zero. By randomly adding the UPC of unknown appliances to the known appliances every day, sequences of known and unknown appliances are decoupled. The hypothesis is that NILM algorithms can no longer learn these sequences, making the algorithms potentially more variable and helping them to become more generalizable for transferability. This means, for example, that if in the real data a kettle (for which ground truth data is available) is always activated before a coffee machine (for which no ground truth data is available), a NILM algorithm can learn this dependency. If this sequence is broken in the synthetic data because the UPC is randomly drawn from the unknown appliances, the NILM model trained on this data could become more variable than the model trained on the real data.

All generated datasets, as well as the real-world data, were converted to h5 format using a converter allowing NILMTK to work with the synthetic data.

5.3. NILM disaggregation algorithms

The synthetic datasets are generated to increase the robustness of ML models. Since no stochastic metric has yet been established to reliably compare data, we are convinced that it is not sufficient to compare data using this method (see Section 2). Consequently, we evaluate their quality using disaggregation algorithms from the publicly available NILMTK-contrib repository [31].

For this work, we choose several NILM algorithms. First, the *sequence-to-point* (seq2point) learning which is a state-of-the-art disaggregation algorithm for transferability [35]. The seq2point model from NILMTK is a neural network that builds a regression map from the aggregated power consumption data to the corresponding target appliance sequence [44]. The model maps sliding windows over the aggregated power consumption data to power consumption segments of appliance loads. This technique can be used to match patterns observed in the aggregated power consumption data with the trained characteristics of an appliance.

Furthermore, we use the *sequence-to-sequence* (seq2seq) learning model, a neural network that works quite similarly to the seq2point model [44]. The difference is that it does not output a sequence of data, but only the midpoint of the time window of the disaggregated appliance. The idea is that this midpoint should have a strong correlation to the information of the same point in time in the aggregated power consumption data of the household.

We also test the *Denoising Autoencoder* (DAE), a special deep neural network [65]. It considers the aggregated energy data of a household as a noisy representation of the power consumption of the appliance and tries to remove the other appliances from the aggregated energy data.

In addition, we use *Recurrent Neural Networks* (RNNs). They use LSTMs (Long Short-Term Memory) and therefore can remember information for longer periods of time, making them particularly suitable for sequential data [65]. The RNN should be able to recognize the power consumption profiles of individual appliances in the total power consumption of a household, as well as the amount of appliances in use.

The last state-of-the-art algorithm we test is the *Window Gated Recurrent Units* (windowGRU). It is a neural network that was developed based on RNNs and is similar in structure, but the LSTM units of the neural network were replaced by Gated Recurrent Units (GRU) [66]. The goal was to reduce redundancies and also computational overhead without affecting the performance. The windowGRU is the most computationally intensive algorithm presented in this work.

We selected the presented algorithms because they are state of the art and widely used in energy applications (seq2point and seq2seq [26,44,67,68], DAE [69–71], RNN and GRU [61,72–74]), but also in various contexts such as dynamic network structure reconstruction (DAE) [75], missing data handling in healthcare (DAE) [76], or security threat detection in smart contracts (RNN) [77].

Table 7
Statistical comparison of real and synthetic datasets.

hh	Dataset	Mean	Median	Min.	Max.	SD
*hh*01	noUPC	69.28	0.00	0.00	3006.60	261.59
	ndUPC	327.60	253.44	0.00	4110.11	362.40
	mUPC	285.76	176.62	0.00	7019.94	413.31
	rUPC	285.90	181.23	0.00	6924.82	418.57
	real	298.66	203.46	108.87	6728.40	410.31
*hh*04	noUPC	79.80	23.10	0.27	3551.08	207.44
	ndUPC	640.52	545.17	0.33	5686.31	547.98
	mUPC	573.75	389.13	0.50	7995.47	650.11
	rUPC	576.85	395.09	0.40	7970.15	653.50
	real	582.82	418.03	121.55	7669.19	645.33
*hh*07	noUPC	231.82	260.30	1.80	4494.90	255.87
	ndUPC	713.74	669.49	1.83	5445.09	448.66
	mUPC	682.89	596.38	2.32	8045.83	499.40
	rUPC	682.89	590.73	2.29	8194.56	505.81
	real	687.85	531.63	304.42	6823.92	478.06
*hh*14	noUPC	78.02	1.81	0.50	4600.96	361.77
	ndUPC	885.61	744.93	0.50	9775.47	813.15
	mUPC	795.75	431.67	0.60	8845.92	953.86
	rUPC	795.69	430.53	0.60	8774.77	955.93
	real	792.25	431.29	161.41	8639.39	948.91
*hh*15	noUPC	133.19	39.34	2.10	4767.07	345.72
	ndUPC	759.58	669.77	2.19	6744.92	597.80
	mUPC	723.07	550.69	13.09	7797.44	660.86
	rUPC	722.89	550.83	36.94	7647.72	653.35
	real	724.69	556.79	163.76	7832.91	656.08

5.4. Experimental environment

In this work, we want to determine the influence of UPC on the training of ML models, so we need to be able to compare different datasets fairly. Therefore, we set up an experimental environment to reliably compare the generated synthetic datasets with each other and with the real data.

First, we statistically compare the different synthetic datasets with each other and with the real data. For this purpose, we examine the mean, median, minimum (Min.) and maximum (Max.) power consumption and the standard deviation (SD) of the aggregated energy data of a household. We then evaluate the different synthetic datasets by training each NILM model type (seq2point, seq2seq, DAE, RNN, and windowGRU) 100 times on each dataset and testing each algorithm on all test households. We first evaluate the *noUPC* dataset to establish a baseline

Table 8
Single pattern type — results of a single pattern type appliance models trained and validated on real and synthetic datasets; tested on real data (kettle, average MAE watt, 100 exp.).

	Test hh	noUPC	ndUPC	mUPC	rUPC	real
S2P	*hh*08	277.81	303.93	24.52	27.97	39.08
	*hh*11	99.06	84.00	9.33	10.80	17.26
	*hh*17	81.03	73.55	9.64	10.32	16.73
	MAEUH	152.63	153.83	**14.50**	16.36	24.36
S2S	*hh*08	268.93	311.15	22.81	27.12	33.98
	*hh*11	85.26	75.65	9.33	10.71	15.69
	*hh*17	69.35	68.59	9.796	10.05	15.58
	MAEUH	141.18	151.80	**13.98**	15.96	21.75
DAE	*hh*08	255.21	189.75	31.77	32.45	37.81
	*hh*11	48.47	54.53	12.19	12.05	13.66
	*hh*17	47.91	48.87	13.71	13.45	16.34
	MAEUH	117.20	97.72	**19.22**	19.31	22.60
RNN	*hh*08	302.46	265.51	37.30	37.69	48.76
	*hh*11	104.20	83.01	11.99	13.05	20.70
	*hh*17	87.05	68.01	11.12	11.62	18.83
	MAEUH	164.57	138.84	**20.14**	20.79	29.43
wGRU	*hh*08	112.89	79.39	24.58	23.72	24.59
	*hh*11	47.01	26.77	9.10	10.03	12.22
	*hh*17	45.06	26.98	10.03	10.10	11.04
	MAEUH	68.32	44.38	**14.57**	14.62	15.95

for the synthetic data, then the *ndUPC* dataset to determine whether the results of the NILM models improve when we add some synthetic normally distributed UPC. Next, we evaluate the *mUPC* dataset to examine whether the results improve when we assign more realistic UPCs consecutively by day. Finally, we evaluate the *rUPC* dataset to determine whether the results improve when the more realistic UPC is randomly assigned.

After comparing the synthetic datasets, we evaluate the real data to determine if the synthetic data is capable of training NILM models as well as the real data (see Section 6.2). To compare the results, the five NILM algorithms were trained on the synthetic data from the *mUPC* dataset and compared to models trained on real data.

All experiments were implemented using the NILMTK experiment API and for seq2point, seq2seq, DAE and RNN models with a sample rate of 2; windowGRU models with a sample rate of 8, as otherwise the experiments would not be feasible because they would require too much time and too many resources to perform this study to the required extent (for this reason, other studies have omitted the windowGRU [43]).

On a Tesla V100-PCIE-32 GB GPU and a tensorflow-gpu version 1.15.0, training and validating one epoch of a seq2point model takes 36 s, while one epoch of a windowGRU model takes 1210 s, which is about 33.6 times slower (with sample rate 2; training on 1 903 320 samples, validating on 335 880 samples).

6. Evaluation

Now that we have defined the environment and constraints for our experiments to determine the impact of unknown appliances in smart meter data on training robust ML models for transferability, we can run them and evaluate the results. In this section we present the results of the statistical comparison of the synthetic datasets as well as the results of the comparison using the NILM algorithms (see Section 6.1). We also compare the models of the synthetic dataset that provide the best results for the NILM models with the models trained on real data (see Section 6.2).

Table 9
Multi pattern type — results of multi pattern type appliance models trained and validated on real and synthetic datasets; tested on real data (washing machine, av. MAE watt, 100 exp.).

	Test hh	noUPC	ndUPC	mUPC	rUPC	real
S2P	*hh*03	13.82	18.83	13.34	17.93	13.46
	*hh*05	62.97	47.08	23.60	23.17	23.73
	*hh*06	22.67	26.12	19.70	20.66	20.63
	*hh*09	63.47	28.14	27.13	28.30	26.76
	*hh*13	33.93	53.11	33.69	33.16	27.90
	*hh*19	77.65	66.79	33.65	37.90	39.71
	*hh*20	31.71	11.85	15.90	16.34	16.15
	MAEUH	43.75	35.99	**23.86**	25.35	24.05
S2S	*hh*03	13.91	23.99	16.35	22.79	18.59
	*hh*05	50.30	44.44	20.99	22.38	22.64
	*hh*06	25.87	26.50	17.39	19.72	17.63
	*hh*09	48.61	29.56	29.91	31.01	29.69
	*hh*13	37.85	54.21	34.69	35.81	28.43
	*hh*19	71.48	68.23	30.96	32.53	34.98
	*hh*20	21.29	13.74	15.91	16.30	15.05
	MAEUH	38.47	37.24	**23.74**	25.79	23.86
DAE	*hh*03	38.86	29.99	25.16	25.43	25.44
	*hh*05	91.19	45.61	32.79	30.79	32.88
	*hh*06	54.89	27.41	26.71	27.08	26.45
	*hh*09	58.55	28.44	25.05	25.79	25.59
	*hh*13	67.10	49.77	41.24	42.01	40.36
	*hh*19	134.30	72.79	38.86	38.24	38.64
	*hh*20	32.59	14.11	15.24	16.01	15.88
	MAEUH	68.21	38.30	**29.29**	29.34	29.32
RNN	*hh*03	21.17	24.83	14.47	19.55	13.41
	*hh*05	207.90	56.84	25.64	25.65	24.97
	*hh*06	51.25	28.43	23.19	23.18	22.77
	*hh*09	167.97	28.25	23.48	24.18	24.98
	*hh*13	98.30	65.73	33.25	32.52	31.62
	*hh*19	166.35	80.11	33.97	38.40	45.69
	*hh*20	117.03	12.72	14.90	15.10	14.77
	MAEUH	118.57	42.41	**24.13**	25.51	25.46
wGRU	*hh*03	38.31	27.25	17.57	21.67	22.75
	*hh*05	55.17	50.48	29.03	27.65	30.96
	*hh*06	44.15	25.04	24.22	25.33	27.56
	*hh*09	36.67	29.18	25.44	26.64	27.25
	*hh*13	50.75	49.48	36.94	38.45	38.30
	*hh*19	96.50	70.71	31.84	33.25	34.94
	*hh*20	17.47	14.29	13.30	14.45	15.56
	MAEUH	48.43	38.06	**25.48**	26.78	28.19

6.1. Synthetic data

To determine the quality of the generated synthetic datasets (*noUPC*, *ndUPC*, *mUPC*, *rUPC*; Section 5.2), we first compare them with real data using the standard statistical methods mean, median, minima (Min.), maxima (Max.) and standard deviation (SD) (see Table 7). The *mUPC* and the *rUPC* dataset are statistically most similar to the real data, while the *noUPC* dataset is the least similar.

To further compare all generated synthetic datasets, we trained, validated, and tested the five NILM algorithms seq2point, seq2seq, DAE, RNN, and windowGRU on all four synthetic datasets for both a single pattern type appliance (electric kettle) and a multi pattern type appliance (washing machine).

When we consider the results for the kettle presented in Table 8 and Fig. 5, we observe that the MAEUH results for the models trained on the *noUPC* and the *ndUPC* are by far the worst for all five NILM algorithms investigated. For three of the NILM models tested (DAE, RNN, and windowGRU), the overall results of the *ndUPC* dataset are very highly significantly better than those of the *noUPC* dataset (see Table 10).

The results for the washing machine confirm this observation, where the results of the NILM models trained on the *noUPC* and the *ndUPC* are also the worst (see Table 9 and Fig. 6). But in contrast to the kettle,

Table 10
Single pattern type appliance — significance table with Mann–Whitney-U test (kettle) - ✓= significant improvement - ✓✓= highly significant improvement - ✓✓✓= very highly significant improvement - x = no significant improvement.

	Dataset	noUPC	ndUPC	mUPC	rUPC	real
S2P	noUPC	∘	x	x	x	x
	ndUPC	x	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	✓✓✓	✓✓✓
	rUPC	✓✓✓	✓✓✓	x	∘	✓✓✓
	real	✓✓✓	✓✓✓	x	x	∘
S2S	noUPC	∘	✓✓✓	x	x	x
	ndUPC	x	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	✓✓✓	✓✓✓
	rUPC	✓✓✓	✓✓✓	x	∘	✓✓✓
	real	✓✓✓	✓✓✓	x	x	∘
DAE	noUPC	∘	x	x	x	x
	ndUPC	✓✓✓	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	x	✓✓✓
	rUPC	✓✓✓	✓✓✓	x	∘	✓✓✓
	real	✓✓✓	✓✓✓	x	x	∘
RNN	noUPC	∘	x	x	x	x
	ndUPC	✓✓✓	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	x	✓✓✓
	rUPC	✓✓✓	✓✓✓	x	∘	✓✓✓
	real	✓✓✓	✓✓✓	x	x	∘
wGRU	noUPC	∘	x	x	x	x
	ndUPC	✓✓✓	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	x	✓✓✓
	rUPC	✓✓✓	✓✓✓	x	∘	✓✓✓
	real	✓✓✓	✓✓✓	x	x	∘

Table 11
Multi pattern type appliance — significance table with Mann–Whitney-U test (washing machine) - ✓= significant improvement - ✓✓= highly significant improvement - ✓✓✓= very highly significant improvement - x = no significant improvement.

	Dataset	noUPC	ndUPC	mUPC	rUPC	real
S2P	noUPC	∘	x	x	x	x
	ndUPC	✓✓✓	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	✓✓✓	x
	rUPC	✓✓✓	✓✓✓	x	∘	x
	real	✓✓✓	✓✓✓	x	✓✓✓	∘
S2S	noUPC	∘	x	x	x	
	ndUPC	✓	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	✓✓✓	x
	rUPC	✓✓✓	✓✓✓	x	∘	x
	real	✓✓✓	✓✓✓	x	✓✓	∘
DAE	noUPC	∘	x	x	x	x
	ndUPC	✓✓✓	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	x	x
	rUPC	✓✓✓	✓✓✓	x	∘	x
	real	✓✓✓	✓✓✓	x	x	∘
RNN	noUPC	∘	x	x	x	x
	ndUPC	✓✓✓	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	x	x
	rUPC	✓✓✓	✓✓✓	x	∘	x
	real	✓✓✓	✓✓✓	x	x	∘
wGRU	noUPC	∘	x	x	x	x
	ndUPC	✓✓✓	∘	x	x	x
	mUPC	✓✓✓	✓✓✓	∘	✓	✓✓✓
	rUPC	✓✓✓	✓✓✓	x	∘	x
	real	✓✓✓	✓✓✓	x	✓✓✓	∘

the overall results of all NILM algorithms of the *ndUPC* dataset are very highly significantly better than those of the *noUPC* dataset (see Table 11).

The results for both appliance types strongly indicate that adding UPC to the synthetic smart meter data is essential to obtain robust NILM models that are able to generalize to unseen households.

If we randomly assign the UPC to the days (*rUPC* dataset), the results of all trained models are very highly significantly better in terms of MAEUH compared to the results of the dataset *noUPC* and *ndUPC*. This behavior is observed for the NILM models of both appliance types (kettle see Table 10 and washing machine see Table 11).

These results clearly indicate that it is important for the NILM models that the synthetic data is generated with a UPC in the first place, and also that it really depends on how the UPC is generated. Moreover, the results strongly implicate that it is not sufficient to simply generate the UPC with a normal distribution, but that it is highly beneficial for the transferability of the NILM models that the UPC contains components from the real smart meter data.

For the kettle as well as for the washing machine, our experiments demonstrate that all tested NILM algorithms trained and validated on the *mUPC* dataset provide better results than those trained on the *rUPC* dataset (see Tables 8, 9). For the kettle these results were very highly significant for four of the tested algorithms (seq2point, seq2seq, DAE, RNN), for the washing machine the results were significant for three tested algorithms (seq2point, seq2seq, windowGRU) (see Tables 10 and 11).

In the *mUPC* dataset, the UPC was the same as the UPC in the *rUPC* dataset, but here the UPC of a specific day is added to the action sequence of the same day instead of being added randomly. The results suggest that it is beneficial for ML models to map the UPC to the known appliances on a daily basis, rather than adding them randomly. It can be assumed that the NILM models are able to learn something useful in the sequences of known and unknown appliances. This means that the hypothesis that breaking these dependencies makes the NILM models more variable in terms of transferability (see Section 5.2) does not seem to be true for both tested appliance types.

6.2. Real vs. Synthetic data

Since we discovered that training and validation on the *mUPC* dataset provided the best results for all tested synthetic datasets and NILM models, we compared them to models trained and validated on real data.

For the kettle, the extremely low p-values of the Mann–Whitney U test verify that the differences for all tested NILM algorithms are very highly significant (see Table 10). For the seq2point algorithm, the MAEUH is even improved by more than 73% when training with the synthetic data compared to training with real data (see Table B.12).

Considering Table 8, we notice that all tested models trained on synthetic data (*mUPC*) provide on average lower and therefore better results than models trained on real data. Although the differences are not as substantial as for the seq2point algorithm, the results are nevertheless very highly significant (seq2seq: 58.70%, DAE: 16.75%, RNN: 57.61%, windowGRU: 14.77%, see Tables 10 and B.12).

Comparing the results presented in the Table 8, we notice that the average results of the seq2seq models trained on the *mUPC* are overall the best. The seq2point models improve the most when trained on synthetic data, while the windowGRU models improve the least. This indicates that the windowGRU algorithm is more robust when it comes to overfitting.

Overall, for the kettle, the seq2seq algorithms trained on the synthetic data (*mUPC*) provide the best results (13.98). For the real data the windowGRU models provide the best results (real: 15.95; see Table B.12).

In addition, we evaluated the washing machine (representing the multi PCP type) using the same NILM algorithms and the same data as for the single PCP type (real data and *mUPC* dataset, Table 9). For the washing machine, we were able to observe a very highly significant improvement for the windowGRU (12.49%) when trained on synthetic data (*mUPC*), compared with training on real data (see Tables 11 and B.13). Furthermore, all other tested algorithms also improved when training on synthetic data compared to training on real data (seq2point: 1.03%, seq2seq: 1.67%, DAE: 0.57%, RNN: 3.33%, see Table B.13).

Overall, for the washing machine, the seq2seq algorithms trained on the synthetic data (*mUPC*) provide the best results (23.74). For the

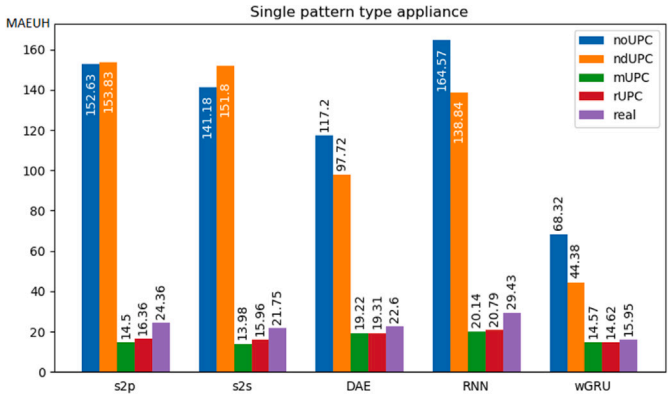


Fig. 5. Comparison of the MAEUH of all tested datasets and NILM algorithms of the single pattern type appliance.

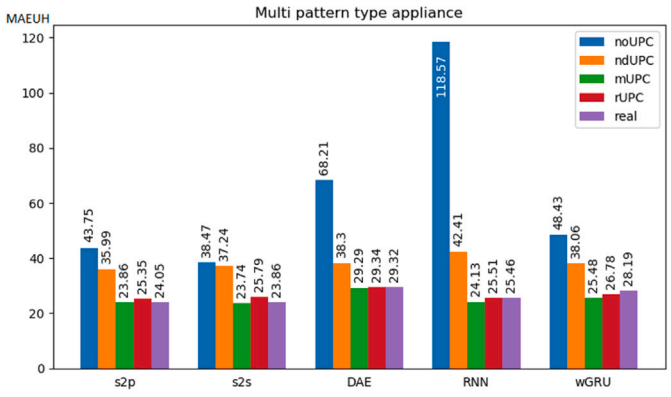


Fig. 6. Comparison of the MAEUH of all tested datasets and NILM algorithms of the multi pattern type appliance.

real data also the seq2seq models provide the best results (real: 23.86; see Table B.12).

In total, during our comprehensive evaluation, we trained and tested five state-of-the-art NILM algorithms (seq2point, seq2seq, DAE, RNN, and windowGRU) and performed over 8500 NILM experiments to test the quality of our synthetic data. The results clearly demonstrate that the performance of state-of-the-art algorithms can be improved by training with synthetic data, if properly designed.

7. Discussion

Synthetic energy data is becoming increasingly important for the development of new applications because it provides many advantages over the use of real data. Theoretically, an unlimited amount of labeled synthetic data can be generated with all available ground truth data, synchronized sampling rates, and no significant differences in available data. Through meaningful synthetic data, ML-based assistance systems can be personalized to households and individuals, allowing these systems to become better trained and therefore more functional, making people's lives more comfortable and safe. However, there are challenges in generating synthetic energy data. It is crucial to ensure that the data is meaningful and represents a domain appropriately.

The presented results demonstrate that transferable NILM models can be meaningfully trained on synthetic data. Considering the first research question of this work, we can definitely conclude that we are able to meaningfully train transferable NILM models on synthetic data when the data is generated with UPC. In certain cases these models even perform significantly better in terms of transferability than models trained on real data. The question remains why this is the case, even though we generated the synthetic data based on real-world data, so they are comparable to a certain extent. However, the method we use to generate the UPC (we subtract all known appliances from the smart meter data) causes spikes in the UPC if the signals from the recorded appliance and smart meter sensors are not perfectly aligned. This means that these spikes occur in the synthetic data, but not in the real data. Interestingly, the spikes appear to be quite similar to the PCPs of the tested single pattern type appliance, where we demonstrated the largest performance improvements for the NILM models. These additional false positive PCPs in the UPC potentially increase the learning challenge for the NILM models and thus may help improve their robustness for transferability. However, this assumption needs to be confirmed by further experiments.

It is not obvious how to reliably evaluate the quality of synthetic data, as we are not aware of any stochastic measure that could be used to do so. Therefore, we decided to evaluate the data with NILM models by comparing their results. Since we noticed during our study that the NILM models provided unstable results even though they were trained with identical parameters on identical data, we developed our methodology and repeated our experiments based on Cochran's sample size. Empirical testing of hypotheses is a common practice in other scientific domains such as physics, and allows us to reliably evaluate the quality of synthetic data using NILM models. It is important to note that NILM models not only provide unstable results when trained on synthetic data, but also when trained on real data (see Section 4.3), allowing our method to also be used to reliably evaluate the quality of real data. We are convinced that our methodology will improve the development of NILM models and the generation of synthetic data. When developing new NILM algorithms and testing their performance, it should always be considered that the model is tested sufficiently often, since only one experiment is not enough to ensure that the algorithm performs better than other state-of-the-art algorithms. Given our findings, it is definitely reasonable to investigate whether it would be possible to develop a NILM model that not only achieves better results, but also more stable results.

Our results demonstrate that this work is essential for understanding how synthetic energy data needs to be structured in order to train meaningful ML models. An key factor in reaching this conclusion was to reliably evaluate the quality of data using NILM models. We are convinced that the results of this work provide a crucial baseline for understanding synthetic data that can be further strengthened. Our methodology demonstrates that it allows to reliably evaluate energy data and also NILM models. This enables the design and deployment of improved energy data and NILM models in the future, which are important for ML-based energy management and optimization systems.

8. Conclusion

In this paper, we focused on determining the effects of unknown appliances in smart meter data on training robust ML models for transferability. In detail, we tested the impact of adding noise, more specifically, UPC, which is the power consumption of all appliances for which we have no separate recordings. The conducted experiments reveal that adding UPC to the synthetic smart meter data is essential to train robust NILM models for transferability tasks on it. However, the experiments demonstrate that it is not sufficient to generate the UPC with statistical methods such as a normal distribution. Furthermore, we demonstrate that the design of the UPC has a high impact on the results. In our case, the UPC which is mapped to the action sequence of the same day, provide the best results. Some dependencies could be learned by the models from the UPC. The fact that these dependencies are broken by the random assignment of the UPC does not seem to have a positive effect on the robustness of the models.

With *SynTiSeD*, we created multiple synthetic datasets using different techniques and methods. To generate useful data, we need to understand what factors affect NILM models and how. To determine

the quality of the generated synthetic datasets, we first statistically compared them to each other. Since the goal of the generated data is to train ML models, we also compared the synthetic datasets to each other as well as to real data through an in-depth analysis by training and validating five state-of-the-art NILM algorithms (seq2point, seq2seq, DAE, RNN, and windowGRU) on each dataset.

We focus our NILM evaluation on appliances that need to be activated by a user, because they represent the vast majority of appliances in a typical household, resulting in two appliance types (see Section 3.1). Since an extensive evaluation with all the necessary details for all instances of both appliance types would be beyond the scope of this work, we examined one appliance each that is representative of its pattern type.

Consequently, we considered one appliance with a single PCP type, namely the electric kettle. We demonstrated that NILM algorithms (seq2point, seq2seq, DAE, RNN, and windowGRU) trained on synthetic data generated with UPC outperform models trained on real data for the kettle (see Section 6.2). Using the seq2point model, we measured an average improvement of 72.62% for the GeLaP dataset (see Table B.12). The improvement of the other algorithms is also highly significant (seq2seq: 58.70%, DAE: 16.75%, RNN: 57.61%, windowGRU: 14.77%).

We further examined a multi pattern type appliance with a complex PCP with multiple programs, namely the washing machine. Also for the washing machine, we prove for the windowGRU that the results of the NILM models trained on the synthetic data generated with UPC are significantly better than the results trained on the real data (12.49% improvement, see Table B.13). The other tested NILM algorithms also provide improved results when trained on synthetic data (seq2point: 1.03%, seq2seq: 1.67%, DAE: 0.57%, RNN: 3.33%). We demonstrated that *SynTiSeD* is able to generate smart meter data including this appliance that allows to meaningfully train NILM models.

Throughout our studies, we observed a high variability of individual results of NILMTK disaggregation models concerning transferability tasks. Therefore, we recommend a methodology to compare these models. The NILM models are trained and validated multiple times with identical settings on the same data. The exact number depends on the expected confidence and is determined using Cochran's sample size calculation (see Section 4). We recommend to repeat at least 100 experiments. To the best of our knowledge, other studies in this domain do not repeat their experiments, which in our experience leads to unreliable results.

We strongly suggest that anyone generating energy data synthetically must consider UPC, because without it, the capability of the data to train NILM models is much worse (see Section 6.1). Furthermore, we suggest that anyone who wants to reliably compare NILM algorithms must consider the methodology we developed, since the models do not produce stable results due to their stochastic nature (see Section 4.2).

By proving that our method for creating and using UPC in combination with *SynTiSeD* is a powerful approach for generating synthetic data that reliably improves the performance of various state-of-the-art NILM models for transferability, we are confident that further research will yield even more useful results. Therefore, in future work, we evaluate more synthetic data generation techniques, especially in combination with UPC, to further improve the performance of ML models.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

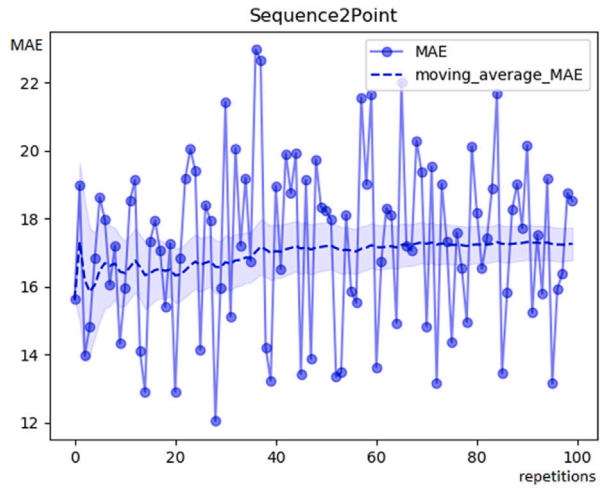


Fig. 7. MAE, Moving Average MAE and confidence interval for the experiment results for a single pattern appliance type (kettle) of 100 seq2point NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

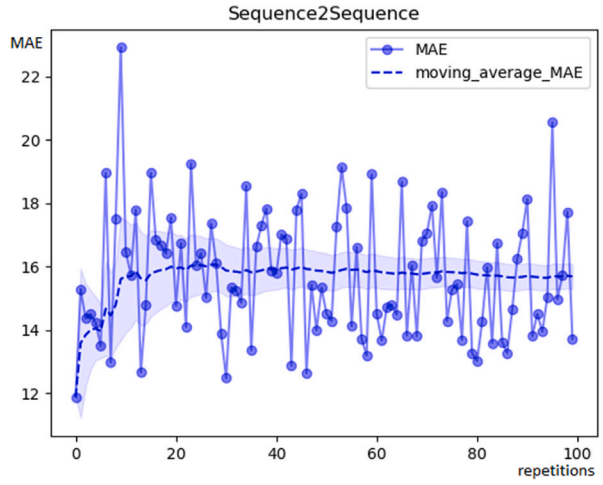


Fig. 8. MAE, Moving Average MAE and confidence interval for the experiment results a single pattern appliance type (kettle) of 100 seq2seq NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

Acknowledgments

The research of Michael Meiser, Benjamin Duppe and Ingo Zinnikus has been funded by the German Ministry for Economics Affairs and Climate Action (BMWK) within the project ForeSightNEXT.

Appendix A

In this section we provide additional graphs including the MAE, Moving Average MAE and confidence interval for the examined single pattern type appliance (kettle, see Appendix A.1) as well as the multi pattern type appliance (washing machine, see Appendix A.2) that illustrate the different results in terms of MAE achieved throughout the 100 experiments for the considered NILM algorithms (Sequence2Sequence, Sequence2Point, DAE, RNN, windowGRU).

A.1. Single pattern type appliance

See Figs. 7–11.

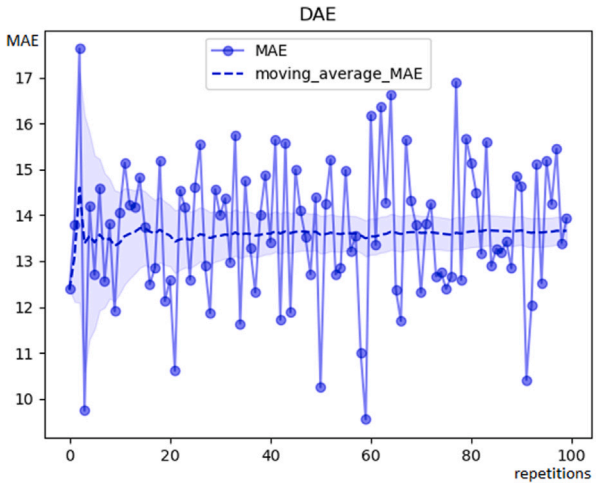


Fig. 9. MAE, Moving Average MAE and confidence interval for the experiment results for a single pattern appliance type (kettle) of 100 DAE NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

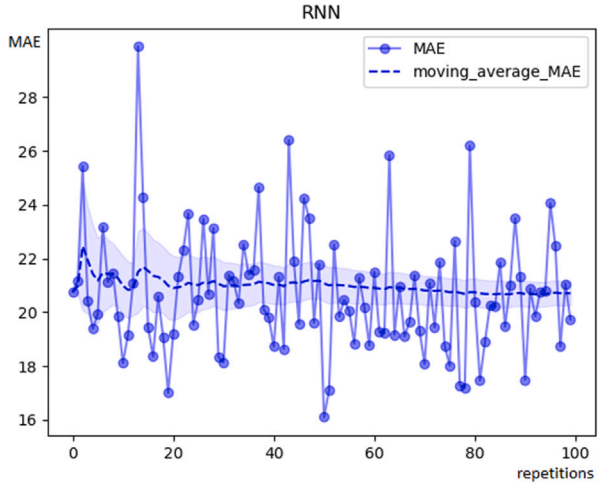


Fig. 11. MAE, Moving Average MAE and confidence interval for the experiment results for a single pattern appliance type (kettle) of 100 RNN NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

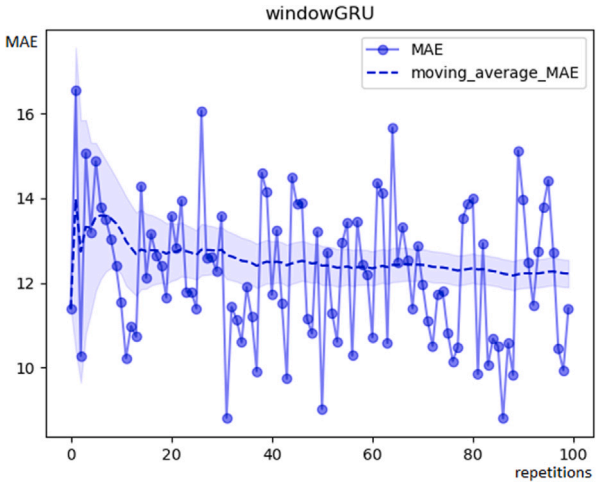


Fig. 10. MAE, Moving Average MAE and confidence interval for the experiment results for a single pattern appliance type (kettle) of 100 windowGRU NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

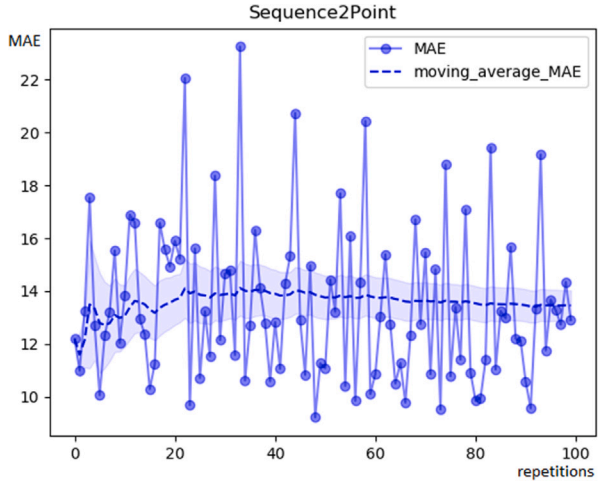


Fig. 12. MAE, Moving Average MAE and confidence interval for the experiment results for a multi pattern appliance type (washing machine) of 100 seq2point NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

A.2. Multi pattern type appliance

See Figs. 12–16.

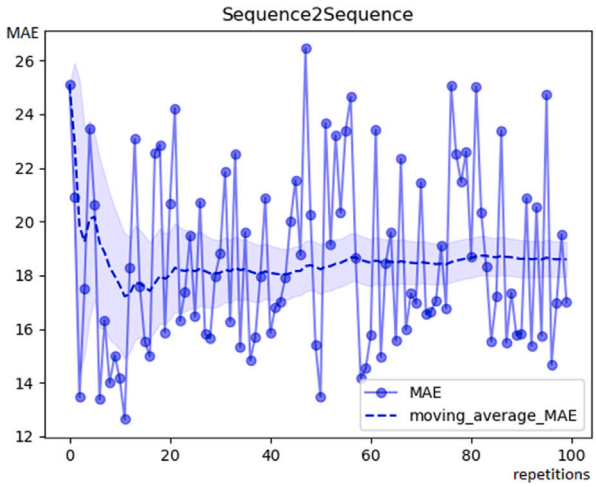


Fig. 13. MAE, Moving Average MAE and confidence interval for the experiment results for a multi pattern appliance type (washing machine) of 100 seq2seq NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

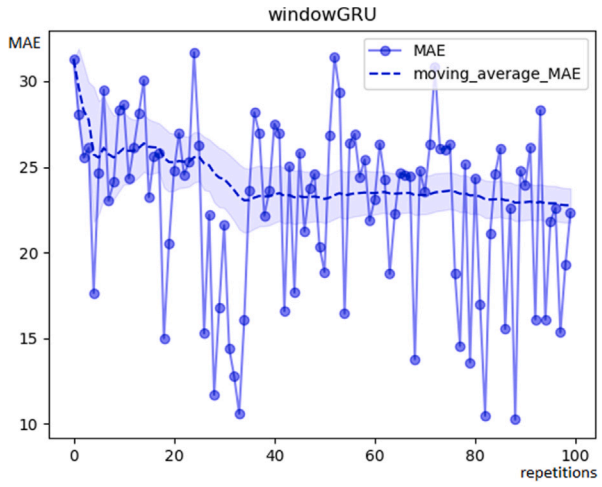


Fig. 15. MAE, Moving Average MAE and confidence interval for the experiment results for a multi pattern appliance type (washing machine) of 100 windowGRU NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

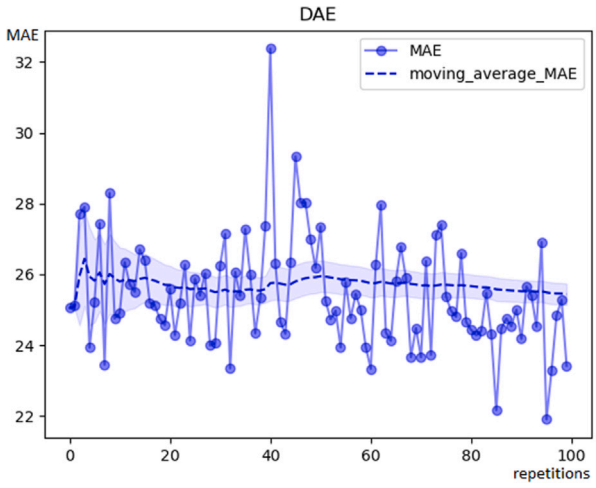


Fig. 14. MAE, Moving Average MAE and confidence interval for the experiment results for a multi pattern appliance type (washing machine) of 100 DAE NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

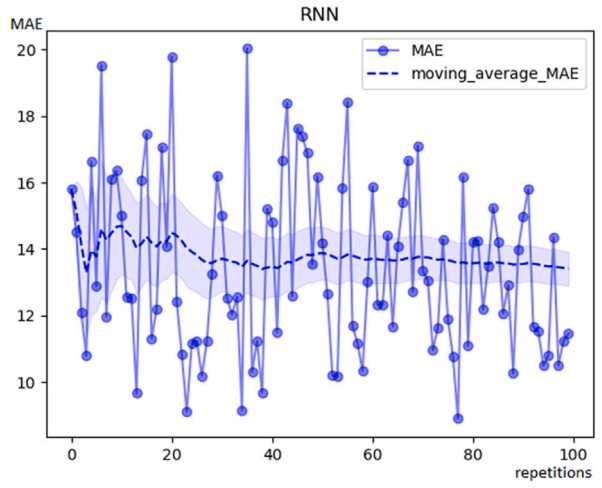


Fig. 16. MAE, Moving Average MAE and confidence interval for the experiment results for a multi pattern appliance type (washing machine) of 100 RNN NILM models trained and tested on the real data of GeLaP dataset (same type of NILM model, same training households and test household, same model parameters).

Table B.12
Single pattern type — comparison of the average MAE results of each household and the MAEUH (watt) of single pattern type appliance models trained and validated on real data and synthetic data (*mUPC*) - (kettle, 100 exp.).

	Test hh	real av. MAE	mUPC av. MAE	Impr. perc.	M-W-U test *p*-value
S2P	*hh*08	39.077	24.515	59.40%	$2.98 * 10^{-34}$
	*hh*11	17.257	9.332	84.93%	$3.26 * 10^{-34}$
	*hh*17	16.731	9.642	73.52%	$3.07 * 10^{-34}$
	MAEUH	**24.355**	**14.496**	**72.62%**	$\mathbf{3.53 * 10^{-32}}$
S2S	*hh*08	33.975	22.813	48.93%	$3.07 * 10^{-34}$
	*hh*11	15.689	9.333	68.11%	$2.72 * 10^{-34}$
	*hh*17	15.581	9.796	59.05%	$3.22 * 10^{-33}$
	MAEUH	**21.748**	**13.981**	**58.70%**	$\mathbf{1.66 * 10^{-31}}$
DAE	*hh*08	37.811	31.767	19.03%	$5.41 * 10^{-21}$
	*hh*11	13.662	12.194	12.03%	$1.23 * 10^{-10}$
	*hh*17	16.339	13.709	19.19%	$4.43 * 10^{-26}$
	MAEUH	**22.604**	**19.224**	**16.75%**	$\mathbf{1.18 * 10^{-13}}$
RNN	*hh*08	48.764	37.304	30.72%	$2.74 * 10^{-25}$
	*hh*11	20.703	11.987	72.71%	$3.36 * 10^{-32}$
	*hh*17	18.829	11.115	69.41%	$1.20 * 10^{-31}$
	MAEUH	**29.432**	**20.135**	**57,61%**	$\mathbf{1.93 * 10^{-28}}$
wGRU	*hh*08	24.589	24.577	0.05%	0.777
	*hh*11	12.216	9.103	34.2%	$2.93 * 10^{-24}$
	*hh*17	11.035	10.026	10.07%	$2.44 * 10^{-6}$
	MAEUH	**15.947**	**14.569**	**14.77%**	$\mathbf{3.19 * 10^{-9}}$

Table B.13
Multi pattern type — comparison of the average MAE results of each household and the MAEUH (watt) of multi pattern type appliance models trained and validated on real data and synthetic data (*mUPC*) - (washing machine, 100 exp.).

	Test hh	real av. MAE	mUPC av. MAE	Impr. perc.	M-W-U test *p*-value
S2P	*hh*03	13.459	13.339	0.9%	0.795
	*hh*05	23.734	23.603	0.55%	0.532
	*hh*06	20.628	19.701	4.7%	0.012
	*hh*09	26.759	27.132	−1.37%	0.116
	*hh*13	27.896	33.694	−17.21%	$9.19 * 10^{-30}$
	*hh*19	39.713	33.653	18.01%	$4.50 * 10^{-27}$
	*hh*20	16.151	15.897	1.6%	0.002
	MAEUH	**24.049**	**23.860**	**1.03%**	**0.890**
S2S	*hh*03	18.587	16.354	13.66%	$4.31 * 10^{-6}$
	*hh*05	22.637	20.992	7.84%	$1.49 * 10^{-18}$
	*hh*06	17.630	17.391	1.38%	0.498
	*hh*09	29.687	29.913	−0.76%	0.295
	*hh*13	28.430	34.689	−18.04%	$3.89 * 10^{-32}$
	*hh*19	34.984	30.958	13.01%	$6.45 * 10^{-32}$
	*hh*20	15.051	15.906	−5.38%	$1.40 * 10^{-5}$
	MAEUH	**23.858**	**23.743**	**1,67%**	**0.863**
DAE	*hh*03	25.437	25.162	1.09%	0.318
	*hh*05	32.877	32.789	0.27%	0.842
	*hh*06	26.445	26.713	−1.01%	0.130
	*hh*09	25.585	25.054	2.12%	0.001
	*hh*13	40.359	41.240	−2.14%	$5.25 * 10^{-6}$
	*hh*19	38.644	38.860	−0.56%	0.351
	*hh*20	15.884	15.237	4.24%	0.0001
	MAEUH	**29.319**	**29.294**	**0,57%**	**0.825**
RNN	*hh*03	13.410	14.468	−7.32%	0.001
	*hh*05	24.970	25.638	−2.6%	$1.95 * 10^{-5}$
	*hh*06	22.771	23.193	−1.82%	0.010
	*hh*09	24.979	23.479	6.39%	$1.88 * 10^{-12}$
	*hh*13	31.618	33.253	−4.92%	$1.47 * 10^{-12}$
	*hh*19	45.686	33.974	34.47%	$2.56 * 10^{-34}$
	*hh*20	14.770	14.902	−0.89%	0.078
	MAEUH	**25.458**	**24.130**	**3.33%**	**0.894**
wGRU	*hh*03	22.752	17.571	29.49%	$1.45 * 10^{-9}$
	*hh*05	30.960	29.027	6.66%	$1.28 * 10^{-7}$
	*hh*06	27.561	24.224	13.78%	$4.83 * 10^{-10}$
	*hh*09	27.246	25.444	7.08%	$1.09 * 10^{-11}$
	*hh*13	38.304	36.937	3.7%	$4.00 * 10^{-5}$
	*hh*19	34.938	31.842	9.72%	$2.10 * 10^{-13}$
	*hh*20	15.558	13.296	17.02%	$7.51 * 10^{-10}$
	MAEUH	**28.189**	**25.477**	**12.49%**	$\mathbf{3.29 * 10^{-10}}$

Appendix B

For completeness, we here summarize additional results from our study. As we discovered that training and validating NILM models on the *mUPC* dataset provided the best results, we compared them to models trained and validated on real data (average MAE for households and MAEUH). We have summarized the results of the comparison in Table B.12 (single pattern type appliance, kettle) and Table B.13 (multi pattern type appliance, washing machine), including the Mann–Whitney-U (M-W-U) test for significance, the improvement (impr.) from using synthetic data compared to using real data (in percent), and a statement of significance.

References

[1] Gao K, Huang Y, Sadollah A, Wang L. A review of energy-efficient scheduling in intelligent production systems. Complex Intell Syst 2020;6:237–49.
[2] Jaciow M, Rudawska E, Sagan A, Tkaczyk J, Wolny R. The influence of environmental awareness on responsible energy consumption—The case of households in Poland. Energies 2022;15(15):5339.
[3] Mariano-Hernández D, Hernández-Callejo L, Zorita-Lamadrid A, Duque-Pérez O, García FS. A review of strategies for building energy management system: Model predictive control, demand side management, optimization, and fault detect & diagnosis. J Build Eng 2021;33:101692.
[4] Cai W, Wang L, Li L, Xie J, Jia S, Zhang X, Jiang Z, Lai K-h. A review on methods of energy performance improvement towards sustainable manufacturing from perspectives of energy monitoring, evaluation, optimization and benchmarking. Renew Sustain Energy Rev 2022;159:112227.
[5] Machlev R, Heistrene L, Perl M, Levy K, Belikov J, Mannor S, Levron Y. Explainable artificial intelligence (XAI) techniques for energy and power systems: Review, challenges and opportunities. Energy AI 2022;100169.
[6] Richter L, Lehna M, Marchand S, Scholz C, Dreher A, Klaiber S, Lenk S. Artificial intelligence for electricity supply chain automation. Renew Sustain Energy Rev 2022;163:112459.
[7] Ahmad T, Chen H, Huang R, Yabin G, Wang J, Shair J, Akram HMA, Mohsan SAH, Kazim M. Supervised based machine learning models for short, medium and long-term energy prediction in distinct building environment. Energy 2018;158:17–32.
[8] Gu F, Chung M-H, Chignell M, Valaee S, Zhou B, Liu X. A survey on deep learning for human activity recognition. ACM Comput Surv 2021;54(8):1–34.
[9] Kaselimi M, Protopapadakis E, Voulodimos A, Doulamis N, Doulamis A. Towards trustworthy energy disaggregation: A review of challenges, methods, and perspectives for non-intrusive load monitoring. Sensors 2022;22(15):5872.
[10] Garcia FD, Souza WA, Diniz IS, Marafão FP. NILM-based approach for energy efficiency assessment of household appliances. Energy Inform 2020;3(1):1–21.
[11] Hart GW. Prototype nonintrusive appliance load monitor: Progress report 2. MIT Energy Laboratory; 1985.
[12] Armel KC, Gupta A, Shrimali G, Albert A. Is disaggregation the holy grail of energy efficiency? The case of electricity. Energy Policy 2013;52:213–34.
[13] Bundesministerium für Wirtschaft und Klimaschutz P. Smart meter-gesetz final beschlossen: Flächendeckender einsatz intelligenter stromzähler kommt. 2023, https://www.bmwk.de/Redaktion/DE/Pressemitteilungen/2023/05/20230512-smart-meter-gesetz-final-beschlossen.html. Accessed: 2023-08-29.
[14] Geelen D, Mugge R, Silvester S, Bulters A. The use of apps to promote energy saving: A study of smart meter–related feedback in the netherlands. Energy Effic 2019;12(6):1635–60.
[15] Shirani F, Groves C, Henwood K, Pidgeon N, Roberts E. 'I'm the smart meter': Perceptions of smart technology amongst vulnerable consumers. Energy Policy 2020;144:111637.
[16] Pereira L, Nunes N. Performance evaluation in non-intrusive load monitoring: datasets, metrics, and tools—A review. Wiley Interdiscip Rev: Data Min Knowl Discov 2018;8(6):e1265.
[17] Long L, Ye H. The roles of thermal insulation and heat storage in the energy performance of the wall materials: A simulation study. Sci Rep 2016;6(1):24181.
[18] Wei S, Jones R, De Wilde P. Driving factors for occupant-controlled space heating in residential buildings. Energy Build 2014;70:36–44.
[19] Ji R, Zhang Z, He Y, Liu J, Qu S. Simulating the effects of anchors on the thermal performance of building insulation systems. Energy Build 2017;140:501–7.
[20] Muhammad Ashraf W, Moeen Uddin G, Muhammad Arafat S, Afghan S, Hassan Kamal A, Asim M, Haider Khan M, Waqas Rafique M, Naumann U, Niazi SG,

et al. Optimization of a 660 MWe supercritical power plant performance—a case of industry 4.0 in the data-driven operational management part 1. thermal efficiency. Energies 2020;13(21):5592.
[21] Ashraf WM, Uddin GM, Farooq M, Riaz F, Ahmad HA, Kamal AH, Anwar S, El-Sherbeeny AM, Khan MH, Hafeez N, et al. Construction of operational data-driven power curve of a generator by industry 4.0 data analytics. Energies 2021;14(5):1227.
[22] Meiser M, Duppe B, Zinnikus I. SynTiSeD–synthetic time series data generator. In: 2023 11th workshop on modelling and simulation of cyber-physical energy systems (MSCPES). IEEE; 2023, p. 1–6.
[23] Ahmed AM, Zhang Y, Eliassen F. Generative adversarial networks and transfer learning for non-intrusive load monitoring in smart grids. In: 2020 IEEE international conference on communications, control, and computing technologies for smart grids (SmartGridComm). IEEE; 2020, p. 1–7.
[24] Kaselimi M, Doulamis N, Voulodimos A, Protopapadakis E, Doulamis A. Context aware energy disaggregation using adaptive bidirectional LSTM models. IEEE Trans Smart Grid 2020;11(4):3054–67.
[25] Chen K, Wang Q, He Z, Chen K, Hu J, He J. Convolutional sequence to sequence non-intrusive load monitoring. J Eng 2018;2018(17):1860–4.
[26] Yang M, Li X, Liu Y. Sequence to point learning based on an attention neural network for nonintrusive load decomposition. Electronics 2021;10(14):1657.
[27] Faustine A, Pereira L, Bousbiat H, Kulkarni S. Unet-NILM: A deep neural network for multi-tasks appliances state detection and power estimation in NILM. In: Proceedings of the 5th international workshop on non-intrusive load monitoring. 2020, p. 84–8.
[28] Kim J, Le T-T-H, Kim H, et al. Nonintrusive load monitoring based on advanced deep learning and novel signature. Comput Intell Neurosci 2017;2017.
[29] Çavdar İH, Faryad V. New design of a supervised energy disaggregation model based on the deep neural network for a smart grid. Energies 2019;12(7):1217.
[30] Toledo-Orozco M, Celi C, Guartan F, Peralta A, Álvarez-Bel C, Morales D. Methodology for the disaggregation and forecast of demand flexibility in large consumers with the application of non-intrusive load monitoring techniques. Energy AI 2023;13:100240.
[31] Batra N, Kukunuri R, Pandey A, Malakar R, Kumar R, Krystalakos O, Zhong M, Meira P, Parson O. Towards reproducible state-of-the-art energy disaggregation. In: 6th ACM int. conf. on systems for energy-efficient buildings, cities, and transportation. 2019, p. 193–202.
[32] Reinhardt A, Klemenjak C. How does load disaggregation performance depend on data characteristics? Insights from a benchmarking study. In: 11th ACM int. conf. on fut. energy sys.. 2020, p. 167–77.
[33] Kelly J, Knottenbelt W. Neural nilm: Deep neural networks applied to energy disaggregation. In: 2nd ACM int. conf. on embedded systems for energy-efficient built environments. 2015, p. 55–64.
[34] Delfosse A, Hebrail G, Zerroug A. Deep learning applied to NILM: is data augmentation worth for energy disaggregation? In: ECAI 2020. IOS Press; 2020, p. 2972–7.
[35] D'Incecco M, Squartini S, Zhong M. Transfer learning for non-intrusive load monitoring. IEEE Trans Smart Grid 2019;11(2):1419–29.
[36] Murray D, Stankovic L, Stankovic V, Lulic S, Sladojevic S. Transferability of neural network approaches for low-rate energy disaggregation. In: ICASSP 2019-2019 IEEE int. conf. on acoustics, speech and signal processing. IEEE; 2019, p. 8330–4.
[37] Yang M, Liu Y, Liu Q. Nonintrusive residential electricity load decomposition based on transfer learning. Sustainability 2021;13(12):6546.
[38] Li Q, Ye J, Song W, Tse Z. Energy disaggregation with federated and transfer learning. In: 2021 IEEE 7th world forum on internet of things (WF-IoT). IEEE; 2021, p. 698–703.
[39] Klemenjak C, Faustine A, Makonin S, Elmenreich W. On metrics to assess the transferability of machine learning models in non-intrusive load monitoring. 2019, arXiv preprint arXiv:1912.06200.
[40] Buneeva N, Reinhardt A. AMBAL: Realistic load signature generation for load disaggregation performance evaluation. In: 2017 IEEE int. conf. on smart grid communications. IEEE; 2017, p. 443–8.
[41] Chen D, Irwin D, Shenoy P. Smartsim: A device-accurate smart home simulator for energy analytics. In: 2016 IEEE int. conf. on smart grid communications. IEEE; 2016, p. 686–92.
[42] Reynaud Q, Haradji Y, Sempé F, Sabouret N. Using time use surveys in multi agent based simulations of human activity. In: International conference on agents and artificial intelligence, Vol. 2. SCITEPRESS; 2017, p. 67–77.
[43] Klemenjak C, Kovatsch C, Herold M, Elmenreich W. A synthetic energy dataset for non-intrusive load monitoring in households. Sci Data 2020;7(1):1–17.
[44] Zhang C, Zhong M, Wang Z, Goddard N, Sutton C. Sequence-to-point learning with neural networks for non-intrusive load monitoring. In: AAAI conf. on artificial intelligence, Vol. 32. 2018.
[45] Wilhelm S, Jakob D, Kasbauer J, Ahrens D. GeLaP: German labeled dataset for power consumption. In: Sixth int. cong. on information and communication technology. Springer Singapore; 2021, p. 21–33.
[46] Firth S, Kane T, Dimitriou V, Hassan T, Fouchal F, Coleman M, Webb L. REFIT Smart Home dataset. 2017.
[47] Beckel C, Kleiminger W, Cicchetti R, Staake T, Santini S. The ECO data set and the performance of non-intrusive load monitoring algorithms. In: Proceedings of the 1st ACM conference on embedded systems for energy-efficient buildings. 2014, p. 80–9.
[48] Monacchi A, Egarter D, Elmenreich W, D'Alessandro S, Tonello AM. GREEND: An energy consumption dataset of households in Italy and Austria. In: 2014 IEEE international conference on smart grid communications (SmartGridComm). IEEE; 2014, p. 511–6.
[49] Shin C, Lee E, Han J, Yim J, Rhee W, Lee H. The ENERTALK dataset, 15 hz electricity consumption data from 22 houses in Korea. Sci data 2019;6(1):193.
[50] Chavan DR, More DS, Khot AM. IEDL: Indian energy dataset with low frequency for NILM. Energy Rep 2022;8:701–9.
[51] Batra N, Gulati M, Singh A, Srivastava MB. It's different: Insights into home energy consumption in India. In: Proceedings of the 5th ACM workshop on embedded systems for energy-efficient buildings. 2013, p. 1–8.
[52] Kelly J, Knottenbelt W. The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. Sci Data 2015;2(1):1–14.
[53] Alberternst S, Anisimov A, Antakli A, Duppe B, Hoffmann H, Meiser M, Muaz M, Spieldenner D, Zinnikus I. Orchestrating heterogeneous devices and AI services as virtual sensors for secure cloud-based IoT applications. Sensors 2021;21(22):7509.
[54] Wooldridge M. Intelligent agents: The key concepts. In: ECCAI advanced course on artificial intelligence. Springer; 2001, p. 3–43.
[55] Irani Azad M, Rajabi R, Estebsari A. Non-intrusive load monitoring (NILM) using deep neural networks: A review. 2023, arXiv e-prints, arXiv–2306.
[56] Yue Z, Witzig CR, Jorde D, Jacobsen H-A. Bert4nilm: A bidirectional transformer model for non-intrusive load monitoring. In: Proceedings of the 5th international workshop on non-intrusive load monitoring. 2020, p. 89–93.
[57] Chen K, Zhang Y, Wang Q, Hu J, Fan H, He J. Scale-and context-aware convolutional non-intrusive load monitoring. IEEE Trans Power Syst 2019;35(3):2362–73.
[58] Athanasiadis C, Doukas D, Papadopoulos T, Chrysopoulos A. A scalable real-time non-intrusive load monitoring system for the estimation of household appliance power consumption. Energies 2021;14(3):767.
[59] Luo J, Liu S, Cai Z, Xiong C, Tu G. A multi-task learning model for non-intrusive load monitoring based on discrete wavelet transform. J Supercomput 2023;79(8):9021–46.
[60] Cochran W. Sampling techniques. Wiley publications in statistics, 2nd ed.. John Wiley & Sons; 1963.
[61] Kim TK. T test as a parametric statistic. Korean J Anesthesiol 2015;68(6):540–6.
[62] Shapiro SS, Wilk MB. An analysis of variance test for normality (complete samples). Biometrika 1965;52(3/4):591–611.
[63] Mann HB, Whitney DR. On a test of whether one of two random variables is stochastically larger than the other. Ann Math Statist 1947;50–60.
[64] Siegel AF, Wagner MR. Chapter 10 - hypothesis testing: Deciding between reality and coincidence. In: Practical business statistics (eighth edit.). 8th ed.. Academic Press; 2022, p. 267–309.
[65] Kelly J, Batra N, Parson O, Dutta H, Knottenbelt W, Rogers A, Singh A, Srivastava M. Nilmtk v0. 2: a non-intrusive load monitoring toolkit for large scale data sets. In: 1st ACM conf. on embedded systems for energy-efficient buildings. 2014, p. 182–3.
[66] Krystalakos O, Nalmpantis C, Vrakas D. Sliding window approach for online energy disaggregation using artificial neural networks. In: 10th hellenic conf. on artificial intelligence. 2018, p. 1–6.
[67] Barber J, Cuayáhuitl H, Zhong M, Luan W. Lightweight non-intrusive load monitoring employing pruned sequence-to-point learning. In: Proceedings of the 5th international workshop on non-intrusive load monitoring. 2020, p. 11–5.
[68] Reinhardt A, Bouchur M. On the impact of the sequence length on sequence-to-sequence and sequence-to-point learning for nilm. In: Proceedings of the 5th international workshop on non-intrusive load monitoring. 2020, p. 75–8.
[69] Garcia FCC, Creayla CMC, Macabebe EQB. Development of an intelligent system for smart home energy disaggregation using stacked denoising autoencoders. Procedia Comput Sci 2017;105:248–55.
[70] Bonfigli R, Felicetti A, Principi E, Fagiani M, Squartini S, Piazza F. Denoising autoencoders for non-intrusive load monitoring: improvements and comparative evaluation. Energy Build 2018;158:1461–74.
[71] Liu P, Zheng P, Chen Z. Deep learning with stacked denoising auto-encoder for short-term electric load forecasting. Energies 2019;12(12):2445.
[72] Xia M, Shao H, Ma X, de Silva CW. A stacked GRU-RNN-based approach for predicting renewable energy and electricity load for smart grid operation. IEEE Trans Ind Inf 2021;17(10):7050–9.
[73] Skrobek D, Krzywanski J, Sosnowski M, Kulakowska A, Zylka A, Grabowska K, Ciesielska K, Nowak W. Implementation of deep learning methods in prediction of adsorption processes. Adv Eng Softw 2022;173:103190.
[74] Skrobek D, Krzywanski J, Sosnowski M, Kulakowska A, Zylka A, Grabowska K, Ciesielska K, Nowak W. Prediction of sorption processes using the deep learning methods (long short-term memory). Energies 2020;13(24):6601.
[75] Huang K, Li S, Dai P, Wang Z, Yu Z. SDARE: A stacked denoising autoencoder method for game dynamics network structure reconstruction. Neural Netw 2020;126:143–52.
[76] Kim J-C, Chung K. Multi-modal stacked denoising autoencoder for handling missing data in healthcare big data. IEEE Access 2020;8:104933–43.
[77] Tann WJ-W, Han XJ, Gupta SS, Ong Y-S. Towards safer smart contracts: A sequence learning approach to detecting security threats. 2018, arXiv preprint arXiv:1811.06632.