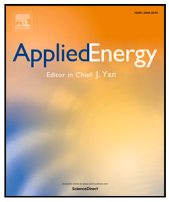
Applied Energy 341 (2023) 121078

Contents lists available at ScienceDirect

Applied Energy

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



An active learning framework for the low-frequency Non-Intrusive Load Monitoring problem

Tamara Todic *, Vladimir Stankovic, Lina Stankovic

Department of Electronic and Electrical Engineering, University of Strathclyde, United Kingdom

ARTICLE INFO

Dataset link: https://pureportal.strath.ac.uk/en/datasets/refit-electrical-load-measurements-cleaned

Keywords:
Active learning
Deep learning
Load disaggregation
NILM

ABSTRACT

With the widespread deployment of smart meters worldwide, quantification of energy used by individual appliances via Non-Intrusive Load Monitoring (NILM), i.e., virtual submetering, is an emerging application to inform energy management within buildings. Low-frequency NILM refers to NILM algorithms designed to perform load disaggregation at sampling rates in the order of seconds and minutes, as per smart meter data availability. Recently, many deep learning solutions for NILM have appeared in the literature, with promising results. However, besides requiring large, labelled datasets, the proposed deep learning models are not flexible and usually under-perform when tested in a new environment, affecting scalability. The dynamic nature of appliance ownership and usage inhibits the performance of the developed supervised NILM models and requires large amounts of training data. Transfer learning approaches are commonly used to overcome this issue, but they often assume availability of good quality labelled data from the new environment. We propose an active learning framework, that is able to learn and update the deep learning NILM model from small amounts of data, for transfer to a new environment. We explore the suitability of different types of acquisition functions, which determines which function inputs are most valuable. Finally, we perform a sensitivity analysis of the hyperparameters on model performance. In addition, we propose a modification to the state-of-the-art BatchBALD acquisition function, to address its high computational complexity. Our proposed framework achieves optimal accuracy-labelling effort trade-off with only 5%–15% of the query pool labelled. The results on the REFIT dataset, demonstrate the potential of the proposed active learning to improve transferability and reduce the cost of labelling. Unlike the common approach of retraining the entire model once a new set of labels is provided, we demonstrate that full re-training is not necessary, since a fine-tuning approach can offer a good trade-off between performance achieved and computational resources needed.

1. Introduction

Since the beginning of the industrial era, human activities are the main driver of climate change — long-term shifts in temperatures and weather patterns, which have now become an existential threat to the world. To mitigate the effects of climate change, the European commission adopted a set of policy initiatives for making Europe climate neutral by the end of 2050. According to these initiatives, net greenhouse gas (GhG) emissions need to be reduced by 55% by 2030, compared to the 1990 levels [1]. To achieve this goal, energy efficiency and optimizing energy consumption in buildings play an important role.

Efficient building energy management requires detailed, fine-grained information related to energy consumption patterns. To extract such information, energy disaggregation, also referred to as Non-Intrusive Load Monitoring (NILM), is an attractive option versus the more expensive and intrusive submetering option to acquire the same data. NILM [2] consists of breaking down the total building's power consumption into its subcomponents, giving insight into how much electricity is consumed by individual electrical devices, and hence, can lead to quantifying the cost of each energy-consuming activity [3], not only in terms of price, but also GhG emissions.

In the early years of NILM research, availability of high-frequency load measurements with sampling frequency of the order of kHz or higher was often assumed (*high-frequency NILM*). However, due to concerns related to user privacy, data storage, and data management, and with the widespread deployment of commercial low-sampling rate smart meters worldwide, low frequency power measurements, with sampling granularity of 1–10 s have become widely available; hence, *low-frequency NILM* used in the supervised setting in [4,5] and unsupervised in [6], has become the main focus of research recently [7]. Low-frequency NILM is fundamentally different to high-frequency NILM.

* Corresponding author.
E-mail addresses: tamara.todic@strath.ac.uk (T. Todic), vladimir.stankovic@strath.ac.uk (V. Stankovic), lina.stankovic@strath.ac.uk (L. Stankovic).

https://doi.org/10.1016/j.apenergy.2023.121078
Received 4 February 2023; Received in revised form 27 March 2023; Accepted 1 April 2023
Available online 10 April 2023
0306-2619/© 2023 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

While the latter, effectively exploits signal transients and harmonic signal analysis to distinguish between multiple appliances and two appliances of the same kind (for example, two air conditioners) [8], low-frequency NILM has to rely on steady-state power measurements only, requiring completely different methodological approaches to address this inverse problem via effective and robust machine learning or statistical data analysis [4].

NILM methods can be *event-based* or *model-based*. In event-based NILM, e.g. supervised [4] and unsupervised [6] graph signal processing approaches, a generic optimization-based approach [9], and an approach using multi-scale wavelet packet tree and ensemble bagging tree [10], events, e.g., the moment an appliance is switched on or off, are detected in the aggregate signal in an unsupervised manner (e.g., using adaptive thresholding as in [11,12]), and then assigned to known appliances by a supervised classifier (e.g., graph signal processing-based [6,13], Support Vector Machine (SVM) [14], Decision Trees (DT) [11], k-NN [15], Dynamic Time Warping (DTW)-based [11]; see [16] for a recent review). In contrast, in model-based NILM, a separate model that takes aggregate measurements as input and consumption or on/off state of an appliance as output, is created for extracting power consumption of each appliance, without relying on prior event detection (see, e.g., Hidden Markov Model (HMM) based methods — the unsupervised approach of [17,18] for incorporating domain knowledge using generic appliance models, [19] for approximate inference in additive factorial HMMs, [20] for signal aggregate constraints in additive factorial HMMs, and spatio-temporal pattern networks [21]). Although event-based approaches are easier to implement and deploy due to data reduction via extraction of events, they rely heavily on accurate edge detection, and hence are, in practice, susceptible to measurement noise and unknown appliances, causing misclassification of appliances with similar operational power range, as reported in [4,6].

To mitigate the aforementioned issues related to errors introduced by edge detection, model-based methods have become dominant in the literature, with deep neural network (DNN)-based solutions growing in popularity. With model-based methods, a separate regression model is typically created for disaggregation of each load, e.g., a separate model for washing machine and another for microwave. Although existing DNN-based NILM algorithms demonstrate good disaggregation performance, there remain major challenges to address before large scale deployment and adoption [7]. Current DNN-based NILM methods require large amounts of labelled data, which is time consuming and expensive to obtain requiring either submetering, reliable time diaries and expert knowledge. Additionally, labelling is not a one-off task due to the large diversity of devices used in different houses/buildings, and *dynamic* nature of these devices – i.e., differing power consumption profiles for the same type of appliance or setting, or power signatures of appliances change over time due to wear-and-tear over the years of usage, or new appliances frequently introduced – all requiring regular re-labelling, as pointed out in [7,22]. This is a significant challenge to existing NILM models, causing them not to work as expected, because conditions (or data statistics) have changed compared to the ones in which the models were developed and trained. In the context of reliability, practicality and maintaining acceptable accuracy in a real-world building setting, a recent review paper [7] suggests continuous learning and active learning methods to develop flexible algorithms, adaptable to changing environments.

Active learning [23] is a modern approach to machine learning, proposed to significantly reduce the amount of data needed to train a model by selecting data samples that would improve its performance the most, via intelligently designed *acquisition functions*. See also a recent review [24]. Active learning can improve transferability of models by labelling only a small portion of data from the new environment, and improve adaptability, by labelling on-the-fly new appliance data [7]. Active learning can boost public trust and confidence when using artificial intelligence algorithms, by implementing a *human-in-the-loop* concept giving domain experts the opportunity to interact with the model by iteratively re-labelling selected samples.

For the more challenging and less investigated time series signal analysis, active learning has been explored in [25] for time series anomaly detection, but without transfer learning. Active transfer learning is proposed in [26] for data forecasting, but active learning is used to choose samples from source domains, with known data distribution both in source and target domains. On the other hand, we consider a more realistic scenario where models need to be adapted and deployed to new environments with unknown data distributions where labelled data is not available, and has to be queried. Active learning for time series classification for applications in driving trajectories learning is presented in [27]. The models used include a SVM and a fully connected neural network, to classify a data point in a latent space for each trajectory. Data used in experiments is balanced for known classes. Although the data is in the time series format, stream-based acquisition functions, that facilitate real-time learning were not considered in [27], and the models were not transferred to a new environment. In our paper, we deal with electricity measurement datasets, which are highly imbalanced and dynamic. Moreover, we explore stream-based acquisition functions suitable for time series data, which have the advantage of online implementation. Furthermore, we demonstrate transfer of a model to a new environment.

There have been a few early attempts to apply active learning to high-frequency, event-based NILM, assuming unrealistic perfect event detection. The first attempt, [28], uses a kNN classifier to assign events extracted from high-frequency load measurements to specific appliances — it is shown that active learning outperforms random selection, but under the assumption of perfect event detection. In [29], a SVM classifier is used, also with high-frequency measurements and perfect event detection assumed. Semi-supervised learning and active learning for NILM are combined in [30], using an RF classifier — the combination outperformed semi-supervised learning alone, but again with high frequency load measurements and perfect event detection assumed. Although a deep learning model is used within the active learning loop in [31], it is still fundamentally an event-based method and assumes perfect event detection with high frequency measurements. These approaches do not take into account errors introduced by the event-detection step prior to NILM classification or load estimation. Furthermore, to the best of our knowledge, there have not been attempts to design active learning methods for low-frequency model-based NILM and testing in a practical scenario with real, dynamic electrical measurements data.

In this paper we propose the first active-learning based method for low-frequency model-based NILM, that can operate at scale using smart meter measurements. As opposed to the research already conducted on active learning for NILM reviewed above, our approach uses low-frequency measurements and model-based NILM method, with a separate model trained for each appliance disaggregated, eliminating the need to introduce impractical assumptions of perfect event detection. In particular, we leverage on the Wave-net NILM approach of [32], as one of the currently best performing models reported in the recent comparative study [33]. We note that though [32] is used to showcase the proposed methodology, other DNN-based NILM solutions, such as deep neural networks from [34], sequence-to-point convolutional neural networks from [35], recurrent neural network from [36], convolutional and gated recurrent unit-based neural networks from [37], a hybrid of a convolutional and a recurrent neural network from [38], or one-to-many CNN architecture from [39], can be used instead with the proposed active learning methodology.

We explore different approaches of selecting the most critical samples to label, i.e., acquisition functions, and discuss their limitations and effect on accuracy and transferability. In the aforementioned active learning approaches — [28] using a kNN classifier, [29] with an SVM classifier, [30] using an RF classifier, and [31] using a DNN, high-frequency, event-based NILM methods are used with classic

uncertainty-based acquisition functions, which yield one data sample at a time. Since DNN methods process a batch of data samples at a time, it is necessary to group the samples before labelling. Creating a batch of samples by simply joining individually queried samples will likely result in samples that are very correlated; this reduces the effectiveness of learning, since for the model to learn more effectively, it is important that it learns from diverse data. For that reason, we explore BatchBALD [40] which can choose a diverse batch of samples but can be computationally demanding [41].

We consider three practical scenarios in terms of availability of labelled data, and analyse how the proposed methods perform in various scenarios. We perform a sensitivity analysis w.r.t pre-set hyper-parameters. We discuss optimal performance-complexity trade-off and determine whether complexity can be reduced without performance loss by not re-training the entire model after each interaction, as is commonly done in existing approaches — re-training a k-NN classifier in [28], an SVM classifier in [29], an RF classifier in [30], or a DNN in [31]. In summary, the contributions of this study are as follows.

- We propose the first active learning framework for model-based low-frequency NILM on real, dynamic data with no assumptions unlike previous literature that assumed perfect edge detection when investigating active learning for event-based NILM for high frequency data.
- We demonstrate how the proposed active learning framework can improve transferability of NILM models to unseen target domains.
- We provide a detailed analysis of the effects of four different querying strategies on performance and transferability, in realistic scenarios, using real household electrical measurements.
- We quantify the trade-off between accuracy and labelling effort and define an optimal trade-off point.
- A modified BatchBALD is proposed to address the high computational complexity of the original BatchBALD acquisition function.
- We show that using fine-tuning after each active learning iteration without re-training the entire model saves on computational time without performance loss.
- We perform detailed sensitivity analysis of the proposed method w.r.t the choice of active learning hyper-parameters, showing that the proposed approach is not sensitive to the number of samples queried but is sensitive to confidence threshold.

The rest of the paper is organized as follows. Background on low-frequency NILM with deep learning, as well as background on active learning with active learning approaches for the NILM problem is presented in Section 2. Section 3 describes the methodology used in this study: the workflow of the proposed method, as well as acquisition functions used. Section 4 provides details of conducted experiments, dataset, DNN-based low-frequency NILM model used, and evaluation methods used. Results are presented and discussed in Section 5, before we conclude in Section 6.

2. Background

In this section a brief background on NILM is first provided, including a review of DNN-based NILM and transfer learning approaches for NILM. Then, active learning is described and the existing work on active learning approaches for NILM is reviewed.

2.1. Low-frequency non-intrusive load monitoring

NILM [2] consists of breaking down the total power consumption of a building into individual loads. That is, the task of NILM is to estimate the power consumption of individual appliances given only the aggregate power consumption. With increased availability of data due to large-scale smart metering roll-out world-wide, low-frequency NILM, where measurements are collected at frequencies below 1 Hz, has been dominant in the recent literature, as observed in recent reviews (see [7] for challenges, methods and perspectives for NILM, [33] for a review of DNNs applied to low-frequency NILM, and [42] for NILM solutions for very low-rate smart meter data) due to practicality and low complexity in terms of data management and communication resources.

Low-frequency NILM is a multi-source separation problem [2] in a very low signal-to-noise ratio environment, and hence is particularly challenging in real-case scenarios, due to many similar loads running in parallel in a house, numerous unknown loads, loads changing over time, and measurement noise. Hence, though introduced over 30 years ago, NILM remains a significant research challenge.

2.1.1. DNN-based low-frequency NILM and model transferability

Numerous machine learning approaches have been used in the past (see Introduction and survey papers such as [43,44]), with DNN-based methods dominating current literature, due to their very good performance (see e.g, [45,46] for comparisons between traditional and DNN-based NILM approaches), flexibility and ease of use (once the models are trained).

A recent review paper [33] summarizes DNN-based low-frequency NILM approaches, concluding that the use of convolutional layers in neural networks has gained in popularity recently — [32] proposes a fully convolutional DNN for a fast sequence-to-point implementation; [37] proposes a CNN architecture, designed to be a generalized network which performs well when transferred to a new domain; [47] proposes a sequence-to-subsequence learning using a CNN; [48] proposes a scale- and context-aware neural network containing convolutional layers; and [49] proposes a CNN and multilabel classification. Recurrent neural network elements in [37,38], and newer concepts, such as generative adversarial networks (GANs) in [47,48] and attention mechanisms in [50,51] have also been attempted. The best performing approaches are the ones using convolutional layers, adversarial losses, multi-task learning and post processing techniques.

Transferability of DNN-based NILM models, i.e., their adaptability to new conditions using user feedback and continuous learning approaches, as well as privacy preserving issues are identified to be key challenges of the current NILM state of the art [7]. The ability to use existing models, or adapt them efficiently to new, unseen environments with dynamic environmental factors and end-user patterns of use, is very important to enable large-scale NILM applications.

Transferability of two DNN architectures across three publicly available datasets — REDD [52], REFIT [53] and UKDALE [54] is explored in [37], without adaptation to new environments. Transferability was successful, though a drop in performance was observed compared to when training and testing with the data from the same dataset. Transferability with adaptation to a new environment is explored in [22], as well as in [51,55] — DNN models are fine-tuned using labelled data from new datasets. In [56], cross-domain and cross appliance transferability is investigated, concluding that if statistics of power consumption are similar between different domains, fine-tuning is not required.

Although transfer learning for NILM has drawn attention recently, many challenges still remain. When transferring a pre-trained DNN-based NILM model to a new environment, the performance is likely to drop significantly. On the other hand, availability of good quality and large amount of labelled data from new domains is assumed when using fine-tuning approaches. In practice, obtaining such labelled data from a new environment requires submetering or manual annotation via a time diary, both of which are resource intensive.

2.2. Active learning

Active learning is used to reduce the amount of data needed to train a deep learning model, with the intuitive understanding that different samples from the dataset have a different contribution to training the model, and if the ones that contribute the most are included in training,

other samples (i.e., those either very similar or of low importance to training) do not have to be, and by excluding them the performance of the model should not decrease significantly [23].

First, a small subset of available data is labelled and included in the training of the initial model. The rest of the data, called "query pool", is not labelled. The initial model makes predictions with the data from the query pool, and for each made prediction, it estimates the certainty associated with the prediction. The more uncertain the model is, the higher the information content of the query pool. Therefore, the samples that the model are the most uncertain about are passed to a domain expert for labelling and included in the training set when returned. Then, the model is retrained using the extended training set. This procedure repeats iteratively until satisfactory performance is achieved. Informativeness of a sample can also be determined independently of the model's output, for example, using distance-based measures. The function used to determine informativeness of a sample, whether it depends on the model output or not, and to choose which samples are labelled and included in training is called "acquisition function" or sampling strategy. The choice of the acquisition function impacts the performance of the active learning model.

If the model decision is made by evaluating all available samples and picking the most valuable of all — then an active learning method uses "pool-based sampling", with the obvious limitation that it cannot be used for near real-time labelling. If the data comes in streams, and each sample is evaluated as it comes and the decision if it should or should not be queried is made at that time, then the active learning method exploits "stream-based sampling" [24].

For deep learning models, forming a batch of individually selected samples may lead to correlated samples inside a batch, which is undesirable. Therefore, batch-aware methods are introduced, such as a state-of-the-art method, BatchBALD [40], demonstrated in [41,57] to have a good performance for many benchmark datasets, but with high computational complexity. It is based on mutual information between batches of samples and model parameters.

2.2.1. Active learning for time series data

Active learning has only recently been introduced for time series data to solve anomaly detection tasks [25,58]. Two publicly available time series datasets used in [25] have significantly shorter lengths of recordings (5 weeks and 7 days, respectively) than the one used in our study (approximately 2 years). Acquisition functions used in [25] include uncertainty sampling, interval random sampling, and top-k sampling based on abnormality score, and a combination (union) of them all. Stream-based sampling was not investigated.

Anomaly detection with active learning and two contrast Variational Autoencoder (VAE)-based models is proposed in [58]. Combination of anomaly scores and standard deviation of posterior distribution at each point for both models is used for choosing samples in the active learning process. However, VAEs do not capture time dependencies in data, which makes them not optimal for time series data. Moreover, this method trains three models in total — two VAEs and one query model which is trained to choose samples based on autoencoders' outputs, which makes the method complex, and hence, training phase is done offline. Our method uses a single DNN model, capable of capturing temporal patterns, that performs both load disaggregation and selection of samples to be queried, which makes it more convenient for full online deployment — for both inference and fine-tuning phases.

An integration of transfer and active learning for time series prediction is presented in [26]. The settings are different to ours — in [26], active learning is used to choose samples from source domains, which are most suitable for transfer to a target domain with *known data distribution*, while we consider a more realistic scenario, when source domain data are labelled and available, and we adapt the model to a new environment with *unknown data distribution*, and labels have to be queried since they are not available in advance. Time series classification with active learning for applications in learning of driving trajectories is presented in [27]. A SVM and a fully connected neural network were used, for classification of a data point in a latent space for each trajectory. Classes used are balanced, while in our paper, highly imbalanced electrical measurements are used. Although data used in [27] is time-series, stream-based acquisition functions suitable for online learning were not considered. In our paper, we explore stream-based uncertainty sampling, which, besides being convenient for time series, has the advantage of online implementation since it does not require the whole query pool to be available in advance.

In contrast to [27], we investigate transfers between different environments, for on/off classification of several types of household devices; we discuss the effect of different query strategies, including stream-based querying, suitable for time-series data; we analyse the best trade-off between labelling effort and accuracy, and also we discuss advantages of fine-tuning over re-training models at each iteration to reduce implementation complexity.

2.2.2. Active learning for NILM

Recent reviews of NILM, including state-of-the-art NILM data sets, feature engineering, as well as learning approaches for NILM are presented in [16,33]. Although new and relevant techniques such as transfer learning and federated learning are discussed, active learning has not been mentioned. This is mainly due to the very limited amount of work on the topic of active learning for NILM with only few initial studies published so far, all focusing on the methodologically very different, high-frequency NILM problem. One of the first attempts to apply active learning to high-frequency NILM [28] uses a kNN classifier trained on BLUED dataset [59] to identify which activation belongs to which appliance. An active learning framework where the algorithm intelligently selects instances for queries based on an informativeness measure, Euclidean distance of the samples in the feature space, is compared to the scenario where the algorithm randomly selects instances to query. The impact of different probability- and distance-based query strategies as well as the choice of the initial training set for event-based high-frequency NILM was investigated in [29]. The performance was evaluated using cross-dataset validation with BLUED dataset [59] and their own ISS kitchen dataset. A combination of semi-supervised and active learning is proposed for training a random forest classifier for event-based high-frequency NILM on high frequency BLUED dataset [59] in [30]. The results show that including active learning outperforms the used semi-supervised learning approach. An active deep learning approach was used in [31], also for an event-based NILM, where a combination of three high-frequency NILM datasets, PLAID [60], WHITED [61] and COOLL [62] with discrete wavelet transform were used to extract high-dimensional appliance features from original current signals.

2.3. Summary

From the above literature review, one can notice that the active learning approaches for high-frequency NILM (sampling rate in order of kHz) yield promising results, but under some impractical constraints, e.g., active learning frameworks for event-based NILM using high-frequency measurements are proposed with the assumption that perfect event detection exists. To the best of our knowledge, there is no work related to active learning for model-based low-frequency NILM, which are more popular now due to their good performance and practicality due to smart metering roll-out, as per [16,33].

We address this gap in this paper by proposing an active learning framework for model-based low-frequency NILM, using WaveNet-based DNN, proven to have good performance and robust to variable duration of patterns of appliances in traditional (i.e., non active learning) settings [32]. No assumptions are made as in above active learning for NILM literature. Furthermore, we explore stream-based uncertainty sampling, which has the advantage of online implementation. We demonstrate the effectiveness of the proposed method on the challenging REFIT dataset [53], which consists of low-frequency load measurements typical of national smart meter roll-outs.

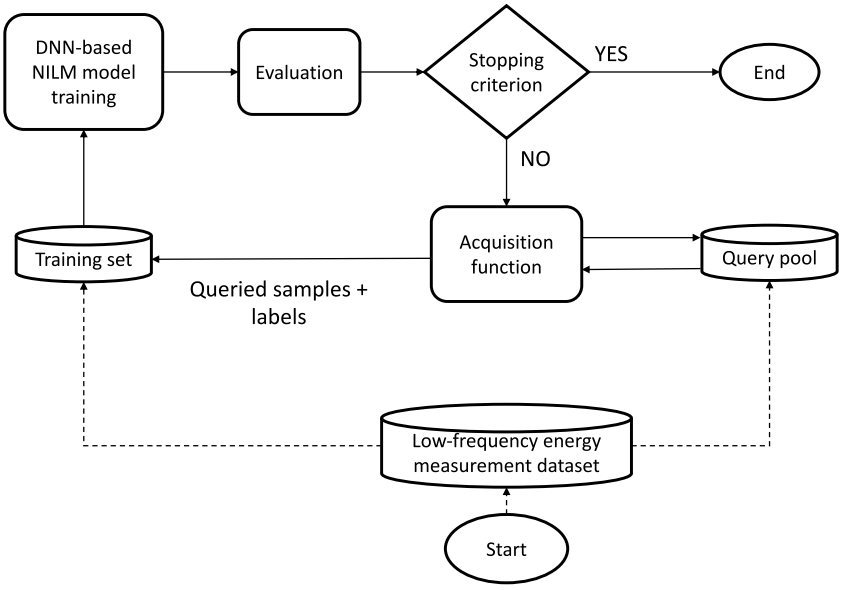


Fig. 1. The workflow of the proposed active learning framework for model-based low frequency NILM.

3. Methodology

In this section, the proposed workflow of the active learning framework for the model-based low-frequency NILM is described. The framework comprises: (i) the formulation of the training set, query pool and testing set, (ii) the various acquisition functions being explored and (iii) the DNN-based NILM model used to showcase the proposed active learning methodology. These are each discussed next.

3.1. Proposed active learning workflow for NILM

As shown in Fig. 1, the dataset is divided into an initial and very small training set ($\boldsymbol{D}_{\text{train}}$), a query pool ($\boldsymbol{D}_{\text{pool}}$) and a test set. Samples from the query pool are considered unlabelled and comprise a representative set of typical on/off samples. A deep learning NILM algorithm is first trained using $\boldsymbol{D}_{\text{train}}$. After the initial training, the obtained model makes predictions on the data from the query pool. The model uses an acquisition function to choose which samples from the query pool should be used for further learning (the set of chosen samples is denoted as $\boldsymbol{Q}$). Having estimated confidence of predictions on data from the query pool, the algorithm queries samples that it was most uncertain about, i.e., the samples that would improve the performance of the model the most, by asking for their corresponding labels. Then these queried samples and their corresponding labels are added to the training data set and they are removed from the query pool. After this step, in the next iteration, the model is trained again with the extended data set that includes newly queried samples. New predictions are made for the samples left in the query pool, and samples that are chosen for querying are added to the training set and removed from the query pool, and so on. This procedure is repeated until all the samples are queried or a stopping criterion met. The stopping criterion can be, for example, the number of queried samples in total, or the estimated achieved accuracy.

3.2. Acquisition functions

Acquisition functions are used to choose which samples to query. The goal is to trade-off between the achieved accuracy and the number of queried samples. The acquisition functions are mainly based on estimated model uncertainty, which is assessed through its output, but other approaches are used as well — for example, the distance of a sample from the other available samples, or a combination of the former two methods. In the following the activation functions considered are described, adapted here to the low-frequency DNN-based NILM problem.

3.2.1. Uncertainty sampling — least confidence

A classification algorithm returns a vector consisting of probabilities of the input samples belonging to each of the classes present in the data set. This vector can be used to assess the confidence of the algorithm in making its prediction by looking at the probability of the predicted class (the highest probability in the vector). If this value is close to 1, then the model is confident about its prediction. Otherwise, i.e., if none of the class probabilities is significantly larger than others, then the algorithm is not confident in its prediction. That is, for each sample, the highest prediction probability among all the classes can be taken as a measure of confidence.

Since the DNN-models usually process multiple points at a time, a pooling function based on a single probability value for the target field (i.e., a set of samples considered at the same time) is defined. In particular, since the state of the target field is considered to be on if it contains at least one sample labelled as on-state (i.e., appliance is on), the maximum probability value of the target field (the maximum among all samples in the field) is used as a confidence measure. This acquisition function is used in pool-based or stream-based sampling fashion, as described next.

- Pool-based sampling
  In pool-based sampling, the algorithm makes predictions on the whole query pool (all samples from the pool have to be evaluated), and then a fixed number of predictions that have the lowest confidence values for the predicted class are queried, expertly labelled and added to the training set. The other samples remain in the pool for querying in the next iteration. An example of pool-based sampling is shown in Fig. 2(a) with query pool set $\boldsymbol{D}_{\text{pool}}$ of ten samples and four samples $\boldsymbol{Q}$ chosen for query. The numbers represent the output of the model, which can be treated as the certainty of the model (its confidence) when making predictions for each sample from the query pool.
- Stream-based sampling
  In stream-based sampling, samples arrive one by one in a sequence. The decision whether a sample should be queried is made by comparing the probability of the predicted class with a predefined threshold — if the probability is lower than the threshold,

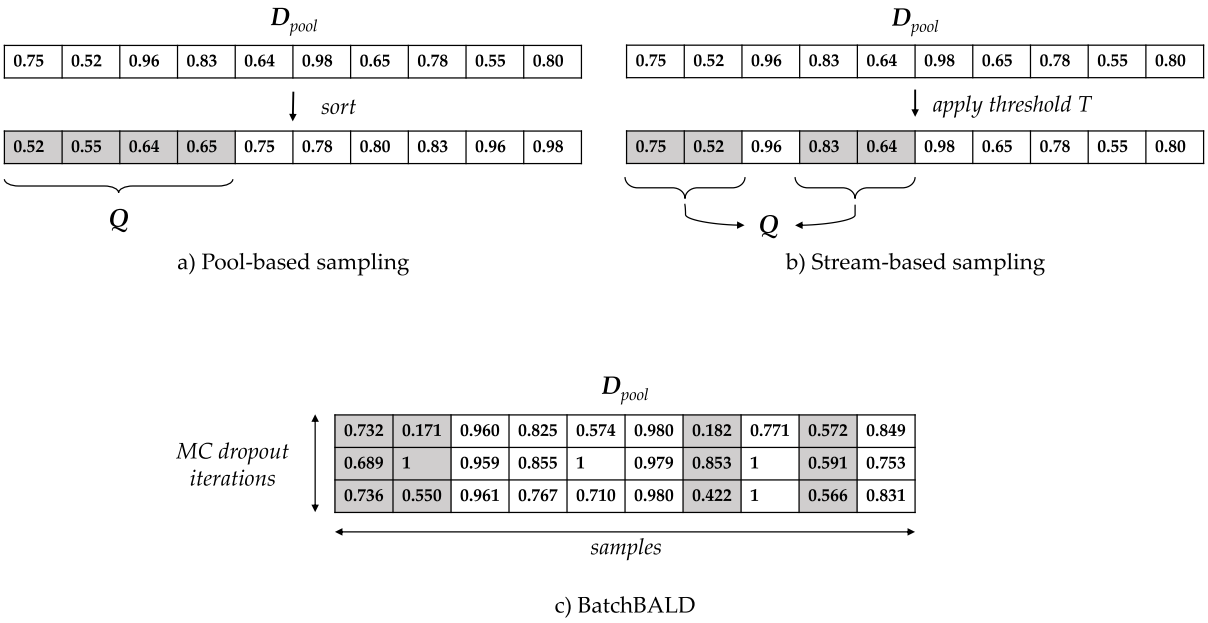


Fig. 2. Pool-based uncertainty (a), stream-based uncertainty (b) and BatchBALD (c) sampling strategy examples. Each number represents the predicted class probability — certainty of the model when making prediction for each sample in $\boldsymbol{D}_{\text{pool}}$. In this example, four samples are queried per one active learning iteration. The used threshold for stream-based uncertainty is $T = 0.9$. The used number of MC dropout iterations (stochastic forward passes) for BatchBALD is 3.

the sample is queried and added to the training set, otherwise, it remains in the pool for querying in the next iteration. An example of stream-based sampling from the same query pool $\boldsymbol{D}_{\text{pool}}$, with threshold $T = 0.9$ and four samples chosen for query $\boldsymbol{Q}$ is shown in Fig. 2(b). If there are less samples than a predefined number of samples whose values fall below the threshold (four in the example in Fig. 2(b)), the active learning process stops, meaning that the model reached high confidence for most the samples.

3.2.2. BatchBALD

Deep learning models typically process a batch of input samples at a time. When using pool- and stream-based uncertainty sampling described above, similarity between chosen samples is not taken into account. If the model is uncertain about one sample, the chances are high that it will be uncertain about other very similar samples (likely to be from the same appliance). That can lead to high redundancy in the chosen samples for querying. In order for learning to progress faster, choosing more diverse batches is necessary.

The BatchBALD [40] acquisition function searches for the optimal batch of samples among all available samples, based on the joint mutual information between the current batch of samples and the model parameters. In this case, the DNN model needs to be Bayesian, which means that its weights are probability distributions instead of single values. This allows estimating model uncertainty based on the variance in the outputs of multiple runs of a model — the greater the variance, the greater the uncertainty of the model, and vice versa. The score of a batch of samples is calculated according to:

$$\begin{aligned} a_{\text{BatchBALD}}(\{\boldsymbol{x}_1 \ldots \boldsymbol{x}_b\}, p(\boldsymbol{\omega} \mid \boldsymbol{D}_{\text{train}})) &= I(\boldsymbol{y}_{1:b}; \boldsymbol{\omega} \mid \boldsymbol{x}_{1:b}, \boldsymbol{D}_{\text{train}}) \\ &= H(\boldsymbol{y}_{1:b} \mid \boldsymbol{x}_{1:b}, \boldsymbol{D}_{\text{train}}) - E_{p(\boldsymbol{\omega} \mid \boldsymbol{D}_{\text{train}})} H(\boldsymbol{y}_{1:b} \mid \boldsymbol{x}_{1:b}, \boldsymbol{\omega}, \boldsymbol{D}_{\text{train}}), \end{aligned} \tag{1}$$

where, $\boldsymbol{x}_{1:b}$ is a batch of b samples drawn from the query pool $\boldsymbol{D}_{\text{pool}}$, $\boldsymbol{y}_{1:b}$ is the corresponding batch of model predictions, and $\boldsymbol{\omega}$ denote the DNN model parameters. I stands for mutual information, H entropy, E mathematical expectation, and p probability density function.

Bayesian approximation for a standard DNN model can be made using the Monte Carlo (MC) dropout technique [63]. Dropout layers are added to the neural network, and multiple stochastic forward passes are simply collected and averaged. The diversity of prediction probabilities of different forward passes reveals how confident the model is about the sample — the higher the variance the lower the confidence. Importantly, the neural network itself remains unchanged. An example of a batch of samples chosen by the BatchBALD algorithm is illustrated in Fig. 2(c). Note that a batch containing a sample with confidence value of 1 can be selected to be queried, if the diversity of the model output is high among the results of different forward passes.

3.2.3. Random sampling

Random sampling, or random query strategy, is the case when a number of samples to be queried is randomly chosen from the query pool — there is no special rule for selecting them, and the model's output for the samples from the query pool is not considered when drawing samples from the pool. This strategy is used as a baseline strategy, and all other strategies which include computing informativeness of samples from the query pool are expected to exceed the prediction performance of the random sampling strategy.

3.3. Low-frequency NILM algorithm

For demonstration purposes, the WaveNet-based NILM approach of [32] is selected, which is highlighted [33] as one of the best performing algorithms for low-frequency NILM. A separate model is created for disaggregating each appliance, which facilitates transferability. One of the model's major benefits is that it has a large field of view. It produces concatenated and processed outputs from multiple layers in the network, each with different fields of view, enabling this model to recognize patterns at multiple scales. Since duration of active use times of loads can vary significantly, this feature is favourable. The algorithm performs binary classification in a sequence-to-point fashion — that is, it slides a window of input aggregate energy consumption measurements to predict whether an appliance is turned on or off at the central point of the sliding window.

4. Experimental setup: Dataset, evaluation metrics and parameter selection

This section provides descriptions of the dataset, evaluation metrics and parameter settings used for demonstration of the proposed methodology. Three experiments are designed to explore the key contributions of this paper. This is followed by the evaluation methodology used to assess the performance of the proposed active learning approach.

4.1. Dataset

A publicly available electrical load measurement dataset - REFIT [53] was used to showcase the active learning methodology in this paper. It consists of whole house, as well as, appliance-level energy consumption measurements from 20 households in the United Kingdom over a period of 2 years, from 2013 to 2015. Energy consumption of 9 individual appliances per house was measured at an 8-sec interval. For this paper, following the example of [32], the data was resampled to a sampling interval of 10 s REFIT has been used widely for benchmarking NILM algorithms (e.g., in [37] for transferability evaluation of two proposed DNN models; in [56] for cross-domain and cross-appliance transfer; in [22] for context-aware load disaggregation; in [55] to address generalization of NILM models; and in [51] where a transformer architecture is proposed for complexity reduction and transferability), and is considered one of the most challenging open access NILM datasets due to the diversity of appliance profiles across houses monitored.

In this paper, appliance models for which active learning is developed are kettle, microwave, toaster and dishwasher, due to their high frequency of use, high consumption, and their presence in most houses. As there is imbalance in the on- and off-time for the 4 appliances chosen, data balancing is performed — when training, i.e., the same amount of on and off samples is included when generating one batch of data samples to mitigate bias. Aggregate measurements expressed in Watts were normalized using z-normalization $Z = \frac{x-\mu}{\sigma}$, where x is the original measurement, μ mean of all measurements in the training data set, σ the standard deviation of all measurements in the training data set and Z is the normalized value. Since the focus is on on/off state classification, to determine whether an appliance was turned on or off, a threshold is applied to the consumption measurements for each appliance, according to Table 1. That is, if the appliance consumption value is above this on-power threshold, then the appliance is considered to be turned on.

4.2. Experiments

1. In the first experiment, we assess whether the active learning approach can be successfully applied to model-based low-frequency NILM when training and testing domains are the same (i.e., the same house is used for training and testing, albeit with different train, query pool and test sets). Practically, in this scenario, only a small set of labelled measurements is available for initial training of the model. For example, this can be achieved using time-diaries for a short period of time, where householders will keep a time-of-use record of their appliances. During the inference-making process, labelling of queried samples can be achieved as follows: via a domain expert and/or the householder will occasionally be asked to confirm when a particular appliance was run, e.g., via an app.
2. In the second experiment, we test whether active learning can enhance the performance of the model when transferred to a new, unseen house. Thus, in practice, time diaries are not needed, since initial training is performed on a publicly accessible dataset. As in Experiment 1, a human will be asked occasionally to label the selected samples from the query pool. We use the data from several houses (excluding the test house) for the initial training set, and the data from the test house for query pool and testing set. Since after each active learning iteration, the model is fully retrained using the initial training samples plus all the samples that were queried, the initial training set has to be kept small. Hence, only few REFIT houses are used for training (see Table 1).

Table 1
On-state power threshold in [W] and training houses in Experiment 2 for each target appliance.

Appliance	Training houses	NAR	On power threshold [W]
Kettle	House 6	0.69	2000
	House 8	0.78	
	House 17	0.58	
Microwave	House 6	0.69	200
	House 8	0.78	
	House 17	0.58	
Toaster	House 6	0.69	50
	House 7	0.58	
	House 8	0.78	
Dishwasher	House 3	0.56	10
	House 6	0.69	
	House 9	0.61	

3. In the third experiment, we use a large pre-training dataset comprising all REFIT houses containing appliances of interest (excluding the test house), instead of a small set of houses as in Experiment 2. When such a large pre-training dataset is used, it is infeasible to perform full retrain of the model after each active learning iteration. Instead, in this experiment, we use active learning together with incremental learning [64] to explore if a larger pre-training dataset combined with fine-tuning the model with the samples queried from the unseen house gives better results than using a smaller pre-training dataset and fully retraining the model after each active learning iteration. It is important to note that complexity of fine-tuning approach does not depend on the size of pre-training dataset, so it can be arbitrarily large — only newly labelled samples are used when fine-tuning, which is not the case with the full-retrain approach of Experiment 2. In this experiment we also test different settings of active learning hyper-parameters — the number of samples queried for a complete active learning iteration for pool- and stream-based uncertainty acquisition, and the confidence threshold value for stream-based acquisition function.

In all these experiments various activation functions are used and their effectiveness is evaluated. The random query strategy is always used as a baseline.

4.3. Parameters

Houses selected for pre-training for each appliance in the second experiment, exploring transferability, are shown in Table 1. The choice was made following the example of [37], and by calculating noise-aggregate ratio (NAR) for all houses, so that there are houses with low, middle, and high NAR present in the pre-training data set, given by:

$$NAR = \frac{\sum_{t=1}^{T} \mid y_t - \sum_{i=1}^{M} x_t^{(i)} \mid}{\sum_{t=1}^{T} y_t}. \quad (2)$$

Here, y_t denotes the total aggregate energy consumption at time instant t, $x_t^{(i)}$ is the consumption of appliance i, T is the monitoring time period, and M denotes the number of known appliances in the house.

REFIT House 2 is chosen for evaluation due to the fact that it is commonly used for testing in NILM literature — [37,56], hence it is suitable for validation and benchmarking. In addition, it contains all the appliances of interest, and has a mid-range NAR of 0.67.

All the parameters used for the training of the DNN, as well as in the active learning loop, are shown in Table 2. The parameters are kept the same as in [32] or are obtained heuristically using the training set. In particular, the input window lengths for kettle and microwave are set to $2^7 - 1$ samples and for dishwasher to $2^{10} - 1$, based on the results reported in [32]. The same window length is set for toaster, since it has similar operation time as kettle and microwave. Target field size

Table 2
Model training and active learning hyper-parameters. 1 sample = 1 window.

Parameter		Value
Input window size	Kettle, microwave, toaster	$2^7 - 1$
	dishwasher	$2^{10} - 1$
Target field		100
Batch size		2^7
Number of maximum epochs		20
Early stopping patience (epochs)		5
Learning rate		10^{-3}
Fine-tuning learning rate		10^{-4}
Number of samples queried per active learning iteration		2^7
Initial training set size for Experiment 1 (samples)		2^{13}
Query pool size (samples)	BatchBALD	2^{12}
	other query strategies	2^{16}
Number of maximum queried samples		2^{14}
Confidence threshold	Exp.1 — microwave & Exp. 2 — toaster	0.95
	all other experiments	0.9
Number of MC dropout iterations		5

of 100 samples is selected as the best performing in [32]. Training is limited to 20 epochs maximum, because of numerous re-training required during the iterative active learning process, and early stopping with patience of 5 epochs is introduced to prevent overfitting. The fine-tuning learning rate is set an order of magnitude lower than the original learning rate used for pre-training, because the weights are already adjusted during pre-training, and although they are tuned, they should not be impacted significantly. In Experiment 3, all trainable network layers are fine-tuned.

The number of samples that are queried for one active learning iteration is kept the same as the batch size used in the training process. Data from the target, evaluation house is split into training set (for Experiment 1), query pool and test set so that each set is a representative set of typical on/off samples from the target house. The initial training set size in Experiment 1 is set to only 2^{13} samples, based on the practical assumption that only a small labelled dataset is available (via a small time-diary); a small initial training set also makes the active learning process feasible, since the initial training set plus queried samples are all used for model training at each active learning iteration. The query pool size is set to 2^{16} samples, to be reasonably larger than the initial training set — to keep the ratio of the labelled and unlabelled number of samples low, and to allow the model to have a variety of samples to choose from, compared to the initial training set. For the BatchBALD acquisition function, the query pool is subsampled to 2^{12} samples, because of the computational demands of the algorithm. The maximum number of queried samples is set to 25% of the whole query pool (i.e., 2^{14}), since this number is sufficient for the performance to stop increasing rapidly (as shown in Section 5), and to keep the time needed for conducting experiments reasonably short. Only for BatchBALD acquisition function, it is set to the whole sub-sampled query pool, considering its size (i.e., 2^{12}).

The confidence threshold for stream-based uncertainty acquisition function is set to 0.9, except for microwave in Experiment 1 and toaster in Experiment 2 it is increased to 0.95, because all the predicted class probabilities are above 0.9 at the beginning of the active learning process, which causes the process to stop without querying any samples. The number of Monte Carlo (MC) dropout iterations that are used in BatchBALD acquisition function is set to 5, which is enough to get a sense of the consistency of model outputs through multiple stochastic forward passes.

Specifications of the PC used for experiments are: Intel(R) Core(TM) i7-7800X CPU @ 3.50 GHz, 32 GB RAM, and a NVIDIA TITAN Xp GPU.

4.4. Evaluation metrics

The performance of the deep learning NILM algorithm is evaluated using F_1 score — the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2} \cdot (FP + FN)}, \tag{3}$$

where TP stands for true positives — when the target sample is positive (appliance on) and the prediction is positive; FP for false positives — when the target sample is negative (appliance off), but the prediction is positive; and FN for false negatives — when the target sample is positive, but the prediction is negative. The performance of the model is measured after each querying iteration on the test data set that is included neither in the training set nor in the query pool.

Active learning results are presented as achieved F_1 score against the percentage of query pool labelled samples. The exact number of queried samples can be obtained having in mind the total query pool size as per Table 2. The optimal point of such a curve is computed as the point that has the minimal Euclidean distance from the "ideal" point whose coordinates are the F_1 score equal to 1 and the number of queried samples equal to 0, calculated according to the following equation:

$$dist = \sqrt{(1 - F_1)^2 + \left(\frac{|\,\boldsymbol{Q}_{\text{total}}\,|}{|\,\boldsymbol{D}_{\text{pool}}\,|}\right)^2}, \tag{4}$$

where the total number of queried samples is denoted by $|\,\boldsymbol{Q}_{\text{total}}\,|$, while $|\,\boldsymbol{D}_{\text{pool}}\,|$ denotes the size of the query pool. This point is considered optimal as the aim is to achieve a high F_1 score while keeping the number of queried samples, i.e., the labelling effort as low as possible. Optimal points are shown as dots in the active learning curves.

The improvement w.r.t the initial model performance – $F_{1\ \text{initial}}$, when none of the samples from the query pool are labelled and added to training, and a gap to the heuristic bound performance – $F_{1\ \text{bound}}$, achieved when the whole query pool is labelled, are calculated according to the following equations:

$$improvement = \frac{F_1 - F_{1\ \text{initial}}}{F_{1\ \text{initial}}}, \tag{5}$$

$$gap = \frac{F_{1\ \text{bound}} - F_1}{F_{1\ \text{bound}}}. \tag{6}$$

It is expected that by adding new samples to the initial training set, the performance will improve. However, the improvement could be negative if the performance drops, due to, for example, adding non-informative samples to the training set from the query pool. On the other hand, the results are expected to be worse compared to the heuristic bound $F_{1\ \text{bound}}$, but the results could exceed this bound, due to, for example, overfitting the model with a very large training dataset, which would lead to the gap being negative.

5. Results & discussion

In this section we present results from each of the three experiments described in Section 4.2. We discuss the performance of active learning, transfer learning of DNN-based NILM models with active learning,

retraining the whole model using the entire training dataset or only fine-tuning using the new labelled samples after each iteration, as well as the effect of different acquisition functions on performance and transferability in a realistic scenario — using real, dynamic household measurements. In addition, we discuss sensitivity to active learning hyper-parameters. All the curves in the plots are smoothed using Savitsky–Golay filter of order 3 and window length 11.

5.1. Experiment 1 Results

The results from the first experiment — demonstrating that active learning can be successfully applied to model-based low-frequency NILM by taking data from a single REFIT house, House 2, for the initial training, query pool and test sets, are shown in Fig. 3. The horizontal axis shows the percentage of samples from the query pool that are labelled, and the vertical axis shows F_1 score achieved by the model. The red dotted line reports $F_{1\ \text{bound}}$, when model is trained on the initial training data set and the whole query pool (100%) together. Those performance bounds are inline with those reported in [32]. The black dotted line shows the initial F_1 score obtained by using the initial training set only.

Note that the experiments were not run until the whole query pool is added to the training data set, but were stopped after 25% of the query pool is added, so the plots show the performance up to that point. For the stream-based uncertainty acquisition function, the active learning process can stop earlier if the stopping criterion is met, i.e., there are insufficient samples with probability of the predicted class below the threshold to form a batch.

The optimal points calculated according to (4) are also marked in the active learning curves for each appliance and query strategy explored in corresponding colours in Fig. 3, showing the best trade-off between labelling effort and accuracy achieved. As expected, the performance of all methods increases with the number of samples added to the training set, and it increases faster for the pool- and stream-based uncertainty acquisition functions than it does for random sampling. Therefore, active learning gives promising results for the training models to disaggregate kettle, microwave and toaster.

It can be seen from Fig. 3, that pool-based and stream-based sampling achieve the optimal performance-complexity point very early (after as little as 5% for kettle and 15% for toaster and microwave, of labelled samples added to the training set), and much before the random sampling baseline except for dishwasher.

For dishwasher there is an increase in performance with samples being labelled, mainly in the range between 1% and 17% of the query pool samples labelled; the increase of random sampling is the same as that of the pool-based strategy, implying that the contribution of all samples in the query pool is similar, or that the pool-based query strategy cannot identify the most informative samples. The stream-based sampling, however, consistently outperforms the other two methods.

Table 3 shows the portion of the query pool that needs to be added to the training set so that the model exceeds 90% of the heuristic bound performance. If 90% was not achieved, the maximum F_1 score and the corresponding portion of query pool are shown. It is worth noticing that with only up to 20% of the query pool samples being labelled and added to the training set, the performance is close to the bound for all appliances, which indicates that the labelling effort could be reduced by as much as 80%. The smallest labelling effort is required for kettle, whose performance is very good to start with, and is of short duration (hence, with a small number of queried samples, many activations can be processed). On the other hand, the most labelling effort is required for toaster and microwave, due to the fact that the model does not disaggregate these two appliances well, as can be seen from the final performance bound, which is around 0.6 for both these appliances. This can also be due to the fact that microwave and toaster have a more statistically complex load profile compared to kettle and are used with different settings, hence more samples are needed to capture the statistics. Interestingly, both pool-based and stream-based sampling achieve similar performance, indicating that off-line labelling is not needed and samples can therefore be labelled as they arrive.

Table 3
Experiment 1: Labelling effort, i.e., % of the labelled query pool samples, $|\boldsymbol{Q}|$, needed to exceed 90% of the bound F_1 score (if possible). The bound F1 corresponds to the results when the entire query set (100%) is used for training.

		Kettle	Microwave	Toaster	Dishwasher
Pool-based	$\lvert \boldsymbol{Q} \rvert / \lvert \boldsymbol{Q}_{\text{pool}} \rvert$	1.6%	10.2%	18.5%	15.8%
	$F_1/F_{1\ \text{bound}}$	92%	90%	90%	90%
Stream-based	$\lvert \boldsymbol{Q} \rvert / \lvert \boldsymbol{Q}_{\text{pool}} \rvert$	1.6%	19.33%	12.1%	8.4%
	$F_1/F_{1\ \text{bound}}$	90%	91%	82%	90%

5.2. Experiment 2 Results

The results of Experiment 2 are shown in Fig. 4. A pre-trained model is transferred to unseen REFIT House 2, and the samples from this house are gradually labelled and added to the training set. The black dotted line represents the disaggregation performance of the pre-trained model on House 2 data without any data from that house added to the training set (0% of the query pool sampled labelled), i.e., before any adaptation to the new environment. The red dotted line reports the heuristic bound F_1 score as in Experiment 1. Even though the query pool for BatchBALD acquisition function is sub-sampled from the original larger pool, curves are shown with respect to the larger pool, to line up the number of queried samples with other acquisition functions.

As can be seen from the plots in Fig. 4, the proposed active learning approach yields promising results for all four appliances tested. As expected, strategically selecting the samples to query significantly improves the performance w.r.t random sampling. Pool- and stream-based uncertainty acquisition functions perform similarly, with pool-based being slightly better for kettle and microwave, and stream-based being slightly better for dishwasher until it reaches high confidence for all samples belonging to the pool. This can also be observed by the optimal points that are reached very early (after only 5%–10% samples labelled). The performance of dishwasher has the steepest increase over a number of iterations. This is expected due to dynamic nature of dishwasher loads within the house — newly added samples provide new information due to variation in dishwasher power patterns over different runs. This is less pronounced with kettle and microwave since newly added samples after 5%–10% of query pool samples being added do not enlarge anymore the informativeness of the training pool. Regarding the toaster, there is a huge jump immediately when fine-tuning is performed due to a large difference between the toaster signature in the target domain (House 2) and those available in the training set. However, after that, the newly added samples do not improve the performance anymore, which can be attributed to the fact that disaggregating toaster is in general very challenging and the results have already come closer to the bound in Fig. 3.

The BatchBALD acquisition function performs similarly to the random acquisition function, which can be explained by the very limited size of the query pool. The BatchBALD acquisition function is very computationally expensive and could not handle a large query pool due to memory constraints. It is not used for dishwasher due to the extremely small query pool size, and hence observed lack of improvement beyond the initial training.

Table 4 shows the improvement w.r.t the initial performance and the gap to the heuristic bound as defined in Eqs. (5) and (6). The best results are shown (maximum performance) within the first 25% of samples added to training, as well as the results with the optimum trade-off points. The results show a high level of improvement for all appliances, bearing in mind that a much higher improvement is desired for lower-performing initial models, i.e., microwave and toaster, since the initial results for kettle and dishwasher were already high. A very small gap for kettle, dishwasher and toaster with pool- and stream-based sampling indicates that there is very little room for improving querying strategies. The optimal trade-off points are generally close to the maximum performance.

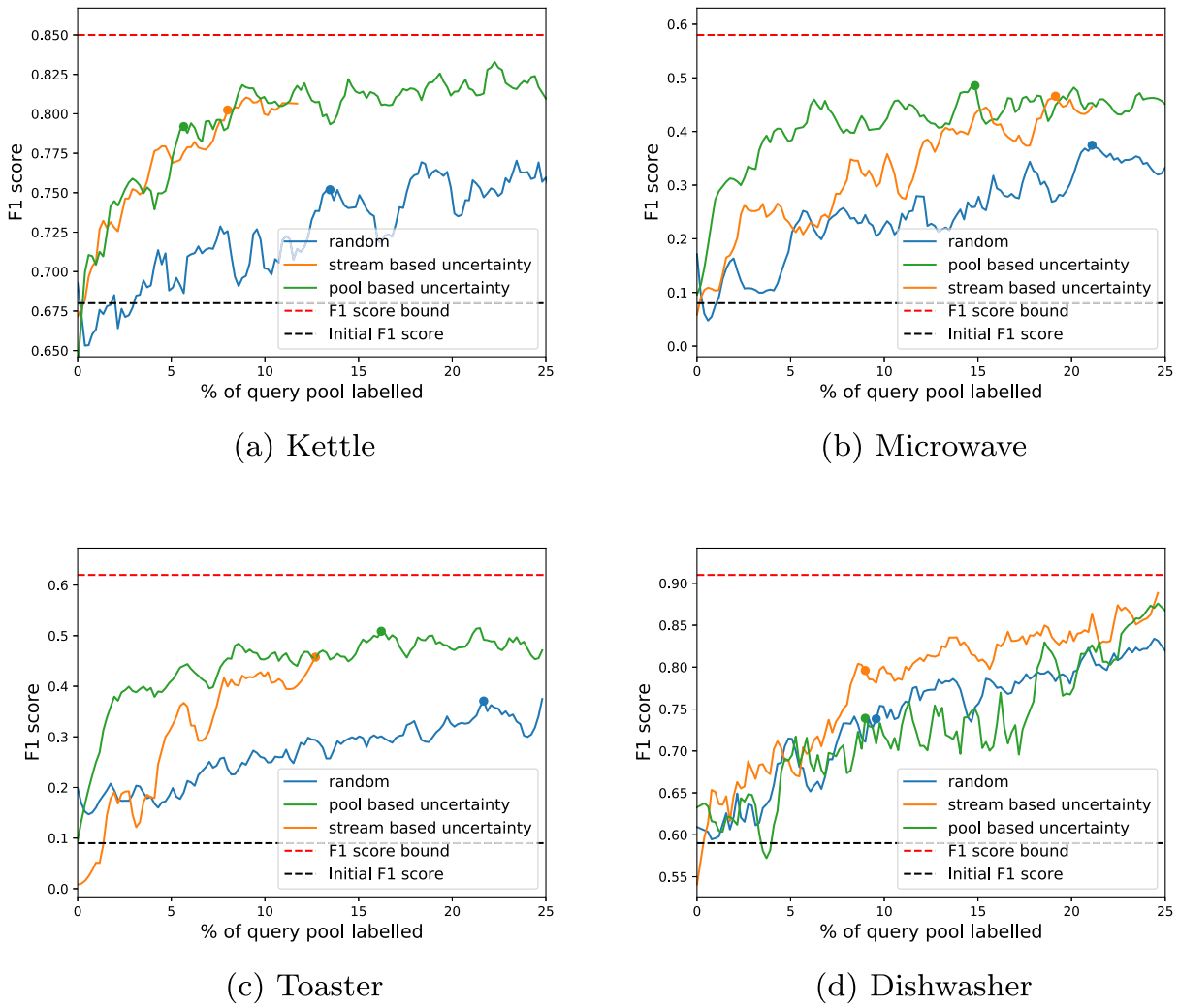


Fig. 3. Experiment 1: Models trained and tested on REFIT House 2 for kettle (a), microwave (b), toaster (c) and dishwasher (d). The red broken line shows the F_1 score bound obtained by using the entire query pool (100%) for training. The dots represent the optimal points obtained using (4). The black broken line is the result obtained with initial training only (0% query pool labelled).

Table 4
Experiment 2: The improvement of the initial performance of the NILM model transferred to a new house using active learning when labelling at most 25% of the query pool, and the gap to the heuristic bound. The results are given for the optimal trade-off point as well as for the best performance.

		Kettle	Microwave	Toaster	Dishwasher
Maximum performance					
Pool-based	Improvement	8.68%	79.23%	104.00%	19.42%
	Gap	−1.02%	14.34%	15.31%	1.95%
Stream-based	Improvement	6.77%	73.63%	122.11%	18.59%
	Gap	0.75%	17.02%	10.14%	2.64%
BatchBALD	Improvement	1.70%	22.51%	72.38%	–
	Gap	5.47%	41.45%	28.43%	–
Optimal trade-off points					
Pool-based	Improvement	7.85%	76.26%	97.71%	13.63%
	Gap	−0.25%	15.76%	17.92%	6.71%
Stream-based	Improvement	6.77%	73.63%	122.11%	13.17%
	Gap	0.75%	17.02%	7.79%	7.09%

Table 5 shows the comparison of F_1 score when initially training the model using data from the same house where the model will be deployed (no transfer), and when a pre-trained model, trained with already available data from multiple houses is transferred to the new house. One can see that both sampling strategies show very small drop in performance when transferred to a new target domain, indicating very fast adaptation due to effectively using the query pool.

Note that the models pre-trained with data from multiple houses can perform better than models trained and tested using data from the same house. This is due to the fact that, as per Experiment 1 settings, initial training set is of very limited size when training and testing with data from the same house, as in a practical scenario, those data will be obtained from time-diaries kept by householders. On the other hand, in Experiment 2, larger amount of data from multiple houses, from an already available, public dataset containing submeter measurements is used, which offers a better variety of data samples for the model to learn.

Considering the presented results of this experiment, it can be concluded that active learning can be used to effectively enhance the performance of pre-trained active learning models when transferred to a new environment, whose appliance profiles (e.g., toaster) are statistically different. Similarly to Experiment 1, stream based sampling shows no performance loss compared to pool based sampling, thereby indicating that online learning is possible.

5.3. Experiment 3 Results

In Experiment 2, after each iteration, when new samples are added to the training set, the entire model is retrained, as is commonly

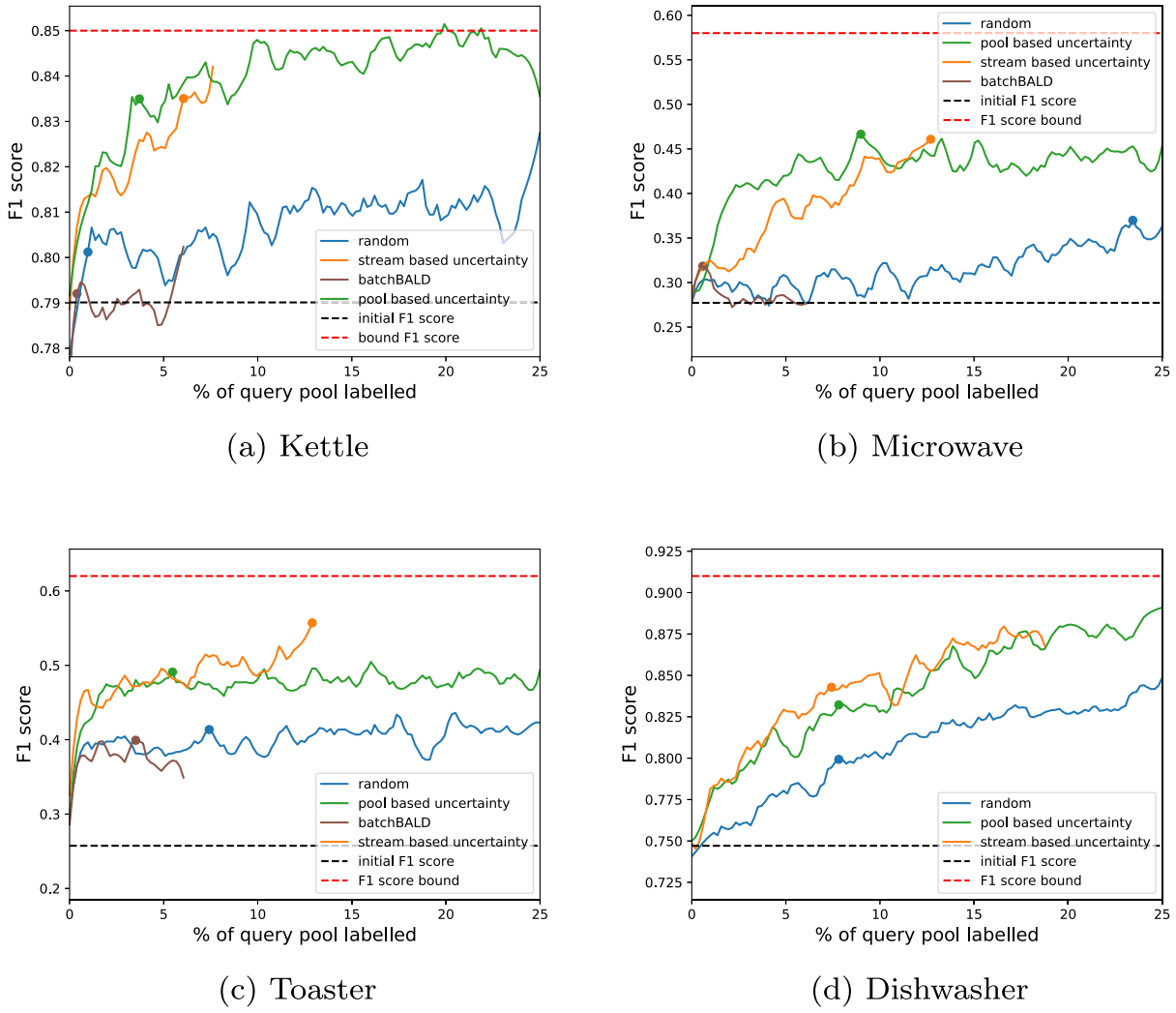


Fig. 4. Experiment 2: Models pre-trained with small dataset transferred to REFIT House 2 for kettle (a), microwave (b), toaster (c) and dishwasher (d). Full retrain of the model is performed in each active learning iteration. The red broken line shows the F_1 score bound as per Experiment 1. The broken black line shows the initial F_1 score obtained using pre-training set only. The dots represent the optimal points obtained using (4).

Table 5
Comparison of the transfer learning results (Experiment 2) and no-transfer learning (Experiment 1) in terms of the maximum F_1 score achieved when labelling at most 25% of query pool. The best results are shown in bold.

		Kettle	Microwave	Toaster	Dishwasher
		Maximum performance			
Pool-based	No-transfer	0.8511	**0.5756**	0.5626	0.8860
	Transfer	**0.8587**	0.4968	0.5251	**0.8922**
Stream-based	No-transfer	0.8241	0.5254	0.5142	0.8897
	Transfer	0.8436	0.4813	**0.5717**	0.8860
		Optimal trade-off points			
Pool-based	No-transfer	0.8217	**0.5591**	0.5501	0.8046
	Transfer	**0.8521**	0.4886	0.5089	**0.8489**
Stream-based	No-transfer	0.8291	0.5254	0.5142	0.8324
	Transfer	0.8436	0.4813	**0.5717**	0.8455

performed in the active learning literature. However, due to these frequent re-training process, the initial training set has to be kept very small, and therefore the execution time to obtain improvements is high. To attempt to mitigate the aforementioned problem, in Experiment 3, we do not retrain the entire model after each iteration, which enables us to increase the size of the initial training set. The results of this experiment — i.e., transfer of a DNN-based NILM model to a new house with a large pre-training dataset and fine-tuning are shown in Fig. 5 and Table 6.

It can be seen from Fig. 5, that the active learning process in this experiment is more stable — active learning curves do not deviate with fine-tuning, especially in the beginning of the process, which is expected since the models are not fully retrained. The optimal trade-off points are again achieved early, with only 5%–15% of added labelled samples, and as observed in previous experiments before, pool-based and stream-based uncertainty sampling lead to similar performance.

In Table 6, gap values are negative both for maximum performance (for all appliances) and optimal points (for all appliances except dishwasher, where it is still very small) when using pool- and stream-based sampling strategies, meaning that bound performance is exceeded, implying that it is worth to use large pre-training datasets and fine-tuning approach.

For toaster, the pre-trained model performs poorly in the new house, but despite that, a higher F_1 score is achieved compared to Experiment 2. Poor initial performance is attributed to the statistical diversity in toaster models, and the fact the House 2 toaster model, and hence load profile, is not available in other houses; however, the active learning approach with fine-tuning overcomes this problem, as shown in Fig. 5(c) and negative gap values in Table 6.

Due to the high computational demands of retraining, BatchBALD in Experiment 2 could handle only a limited number of samples from

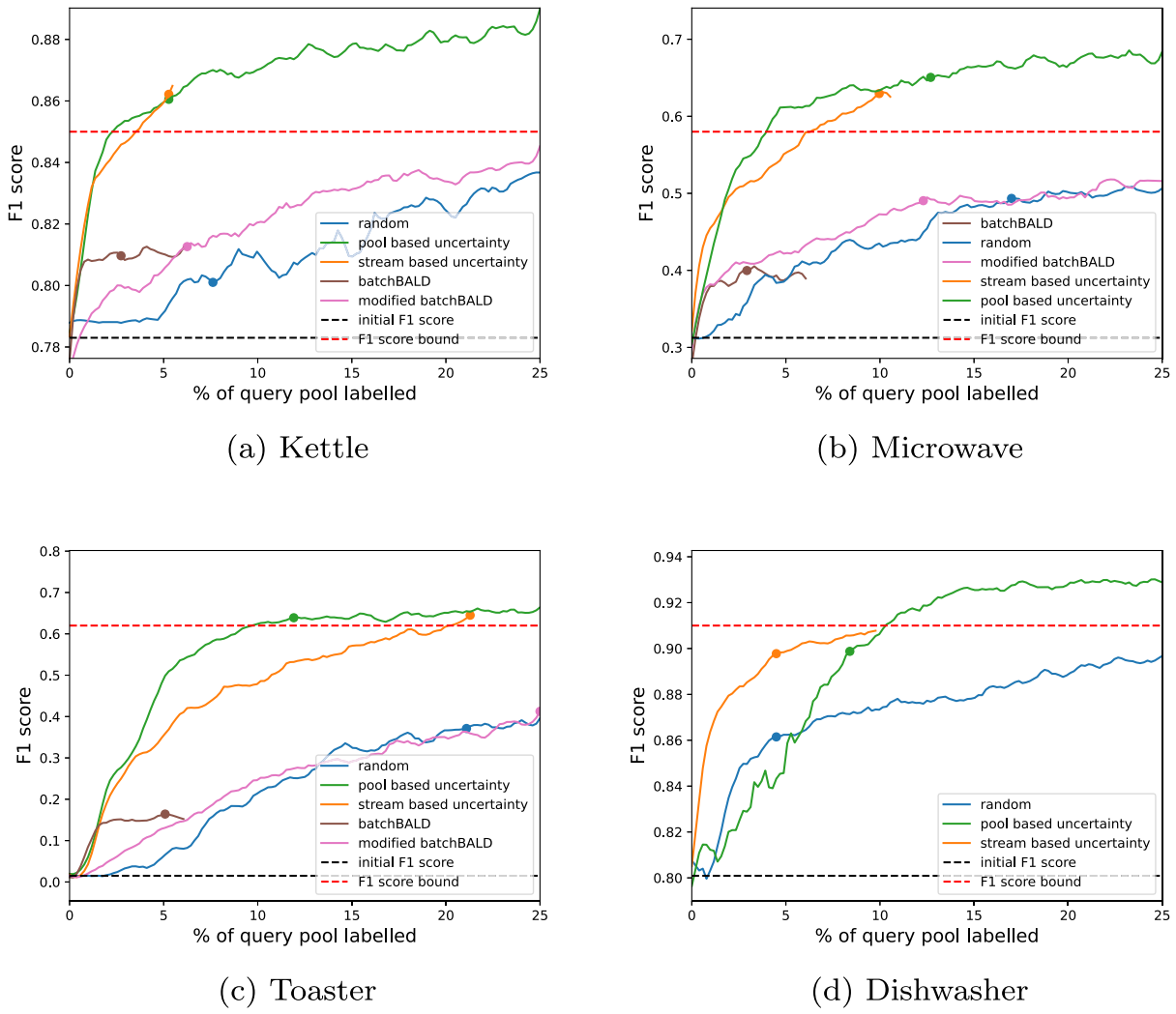


Fig. 5. Experiment 3: Models pre-trained with large datasets transferred to REFIT House 2 for kettle (a), microwave (b), toaster (c) and dishwasher (d). Fine-tuning of the model is performed in each active learning iteration without retraining. The red broken line shows the F_1 score bound as in Experiment 1. The broken black line shows the initial F_1 score obtained using pre-training set only. The dots represent the optimal points obtained using (4).

Table 6
Experiment 3: F_1 score achieved by the NILM model transferred to a new house using the large pre-training dataset and the fine-tuning approach to active learning when labelling at most 25% of query pool.

		Kettle	Microwave	Toaster	Dishwasher
		Maximum performance			
Pool-based	Improvement	12.79%	122.69%	4475.845%	16.84%
	Gap	−4.58%	−19.98%	−9.97%	−2.56%
Stream-based	Improvement	9.71%	103.55%	4226.17%	13.95%
	Gap	−1.72%	−9.67%	−3.97%	−0.02%
BatchBALD	Improvement	3.58%	35.23%	1090.60%	–
	Gap	3.96%	27.14%	71.39%	–
Modified BatchBALD	Improvement	7.26%	71.55%	2752.35%	–
	Gap	0.55%	7.57%	31.45%	–
		Optimal trade-off points			
Pool-based	Improvement	9.62%	116.48%	4243.62%	12.97%
	Gap	−1.64%	−16.64%	−4.39%	0.84%
Stream-based	Improvement	9.71%	103.55%	4226.17%	12.47%
	Gap	−1.72%	−9.67%	−3.97%	1.27%
Modified BatchBALD	Improvement	3.88%	62.30%	2752.35%	–
	Gap	3.68%	12.55%	31.45%	–

the query pool. In this experiment, since re-training is not performed after each label is added, but only fine-tuning, we adapt BatchBALD such that the query pool updates each time a batch of samples is drawn out of it and newly arrived samples are put in the pool to replace the drawn ones. Thus, this could be considered as a hybrid of a pool- and stream-based acquisition and is referred to modified BatchBALD.

The proposed modified BatchBALD method with the introduced adaptation performs better than random sampling for kettle and microwave, compared to the bound performance. In general, BatchBALD performs worse than pool- and stream-based uncertainty sampling, which can be explained by the fact that all samples in the query pool are not highly correlated and it is sufficient to look at their importance and not mutual correlation.

A comparison of full retrain (Experiment 2) and fine-tuning (Experiment 3) in terms of F_1 score is presented in Table 7. Looking at the plots in Fig. 5, and at Table 7, it can be observed that the performance of the model that is pre-trained using a very large dataset and fine-tuned with queried samples reaches higher F_1 score for all appliances tested than the model that is pre-trained using a smaller dataset and fully retrained at each iteration (i.e, Experiment 2).

Using models pre-trained with large datasets and fine-tuning, instead of full retrain, yields the best results among all 3 experiments, with an important benefit that should not be neglected — a significant decrease in time needed for completing the active learning process. An

Table 7
Comparison of full retrain (Experiment 2) and fine-tuning (Experiment 3) — for each appliance the best F_1 score the model achieved when at most 25% of the query pool is labelled.

		Kettle	Microwave	Toaster	Dishwasher
		Maximum performance			
Pool-based	Full retrain	0.8587	0.4968	0.5251	0.8922
	Fine-tuning	**0.8889**	**0.6959**	**0.6818**	**0.9333**
Stream-based	Full retrain	0.8436	0.4813	0.5571	0.8860
	Fine-tuning	0.8646	0.6361	0.6446	0.9102
BatchBALD	Full retrain	0.8035	0.3396	0.4437	–
	Fine-tuning	0.8163	0.4226	0.1774	–
		Optimal trade-off points			
Pool-based	Full retrain	0.8521	0.4886	0.5089	0.8489
	Fine-tuning	0.8639	**0.6765**	**0.6472**	**0.9024**
Stream-based	Full retrain	0.8436	0.4813	0.5717	0.8455
	Fine-tuning	**0.8646**	0.6361	0.6446	0.8984

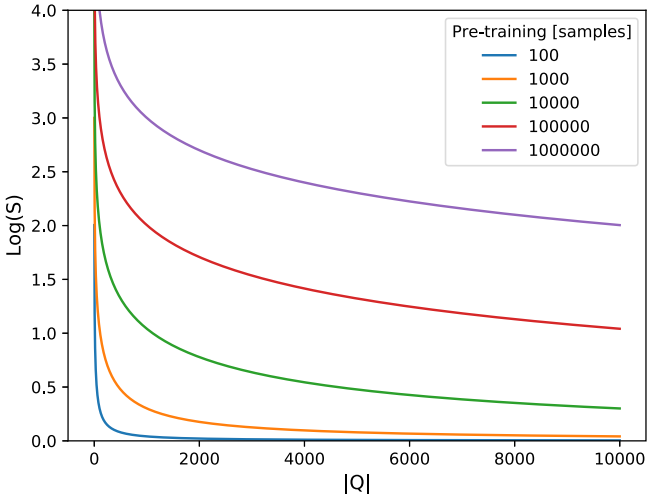


Fig. 6. The speed-up of fine-tuning compared to the full retrain approach to active learning for various sizes of the pre-training dataset (in the number of samples). The horizontal axis shows the number of labelled samples from the query pool.

insight in speed-up that the fine-tuning approach enables is shown in Fig. 6 for various sizes of the pre-training dataset, by using the number of samples included in training as an indicator of time needed for training. The speed-up S is computed as a ratio of samples included in the model training with the full retrain approach (pre-training samples + queried samples) denoted as $|\boldsymbol{D}_{pre-train}|$, and samples included in the model training with the fine-tuning approach (queried samples only, $|\boldsymbol{Q}|$), according to:

$$S = \frac{|\boldsymbol{D}_{\text{pre-train}}| + |\boldsymbol{Q}|}{|\boldsymbol{Q}|}. \tag{7}$$

As it can be seen from Fig. 6, the larger the pre-training dataset, the higher the speed-up of the fine-tuning approach. The fine-tuning approach offers significant time savings, most of which happens in the early active learning process, which is when the model's performance increase is most rapid, as per the results of all aforementioned experiments. Moreover, as mentioned before, with fine-tuning, the size of pre-training dataset can be arbitrarily large, since only the queried samples are used during training.

The results of sensitivity analysis regarding the number of samples queried for one iteration for random, pool- and stream-based uncertainty are shown in Fig. 7. Note that the horizontal axis of the plot shows the percent of the query pool labelled, i.e., the labelling effort. It can be seen from the figure that the performance is not sensitive to the number of queries per iteration.

Results of sensitivity analysis with regards to the confidence threshold used for stream-based uncertainty acquisition function are presented in Fig. 8. A lower confidence threshold leads to more challenging samples added to the training set, and hence faster improvement in performance compared to higher thresholds. On the other hand, a higher confidence threshold implies that more samples are going to be considered, so the process runs for longer. For dishwasher, all confidence threshold levels provide equally steep performance increase, which is likely due to a large number of samples with the confidence level below the lowest threshold (0.9), caused by other loads with similar wattage present in the training dataset, for example, dishwasher is often confused with washing machine [12].

5.4. Results summary

- Active learning can be successfully applied to model-based low-frequency NILM to reduce labelling effort, and to enhance performance of models transferred to new environments.
- Performance of stream-based acquisition function, that can be performed online, is on par with pool-based one that requires presence of the whole query pool in advance and hence cannot be used online.
- Batch-aware acquisition function (BatchBALD [40]) was inferior to other acquisition functions explored, due to its high computational demands. To mitigate the complexity and low accuracy of the original BatchBALD, a modification of it has been introduced.
- Optimal trade-off between accuracy and labelling effort is achieved with 5%–15% of query pool labelled in most of the cases.
- Fine-tuning offers a good trade-off between accuracy and labelling effort and therefore full retrain at each iteration may not be necessary.
- Performance of active learning with pool- and stream-based acquisition functions is not sensitive to the number of samples queried per iteration — same labelling effort yields same performance, but if more samples are queried in one iteration, fewer iterations are required.
- The lower the confidence threshold for stream-based uncertainty acquisition function, the faster the improvement of the model in the beginning of the active learning process; the higher the confidence threshold, the longer the process runs.

6. Conclusions

In order to take advantage of large scale smart meter rollout and NILM to be deployed widely to get itemized electricity consumption reports for improved energy management, it is important to have a way to adapt NILM algorithms to new houses efficiently, to get best-performing algorithms with as little labelled data as possible. This paper demonstrated the viability of active learning to reduce labelling effort, as well as to improve transferability of deep learning models to statistically different and dynamic electrical measurements. Three different experiments were conducted — first, to show that labelling effort can be significantly reduced by using active learning and providing labels only for valuable samples; second, to show that the performance of DNN-based NILM models with active learning, can be enhanced when transferred to a new environment by labelling reasonably small amount of new samples that are informative; and third, to show that full retrain of deep learning models after each active learning iteration may not be necessary – fine-tuning with only newly labelled data from the new environment can produce satisfactory results, offering a good trade-off between performance achieved and computational resources needed.

Different acquisition functions were explored, including pool- and stream-based uncertainty, and batch-aware BatchBALD acquisition function along with a modified BatchBALD to address complexity of

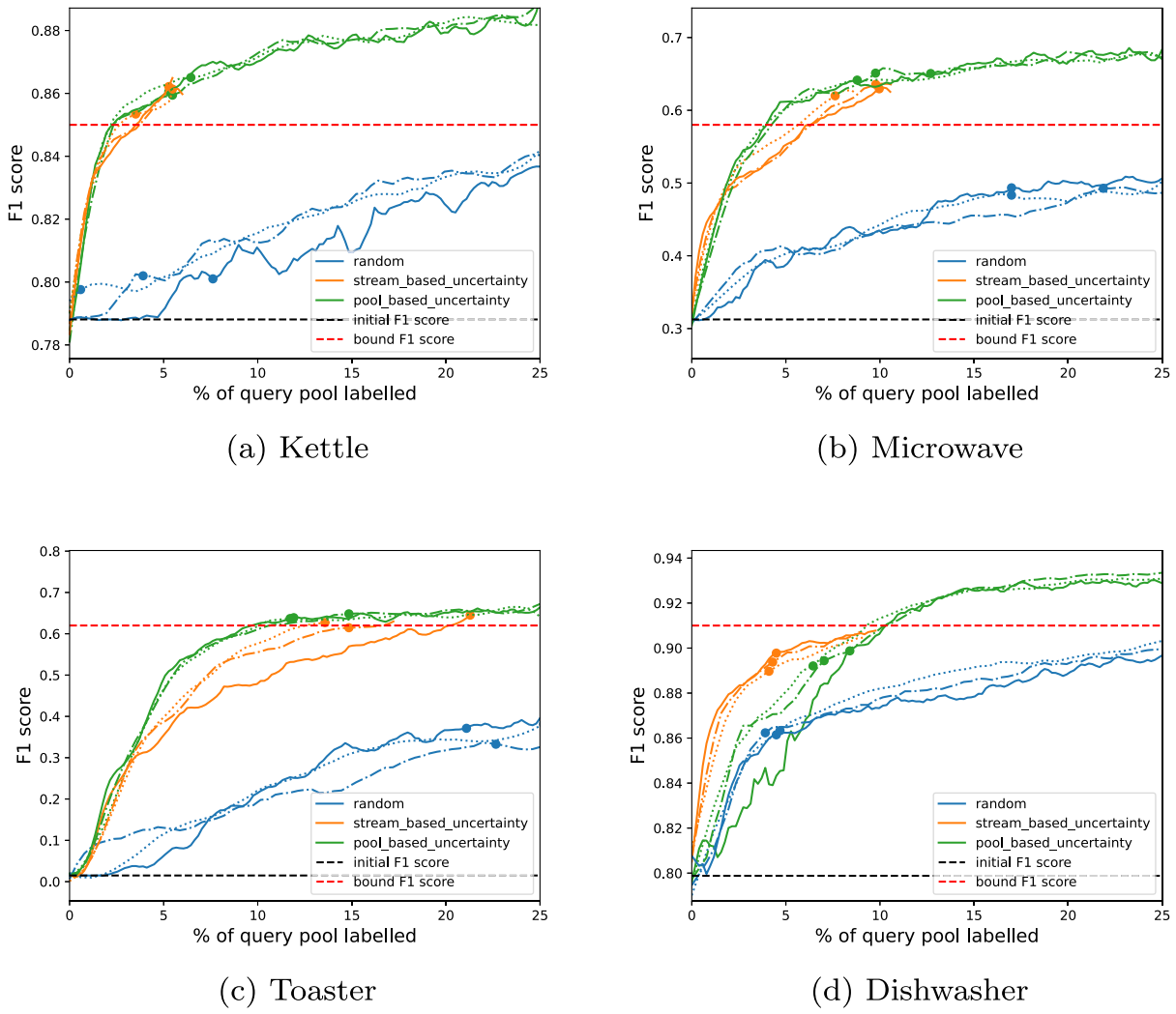


Fig. 7. Experiment 3 — Sensitivity analysis: Models pre-trained with large datasets transferred to REFIT House 2 for kettle (a), microwave (b), toaster (c) and dishwasher (d). Fine-tuning of the model is performed in each active learning iteration with a variable number of samples queried — 128 (solid line), 256 (dash-dotted line) and 384 (dotted line).

the original BatchBALD. Worth noting is that the performance of the stream-based uncertainty, which can be implemented online, was on par with pool-based uncertainty, which requires availability of the whole query pool in advance, and hence cannot be implemented online. BatchBALD acquisition function can consider only small query pool sizes, because of its high computational requirements, and therefore its performance was inferior to other acquisition functions. To overcome this, a modification is introduced to update the query pool in a stream-like fashion, to obtain a hybrid of pool- and stream-based strategy. Though the modified BatchBALD outperformed the original BatchBALD, its performance is still inferior to pool- and stream-based uncertainty strategies. Optimal trade-off between labelling effort and accuracy was discussed — in most of the cases, the optimal point was achieved with 5%–15% of query pool labelled, which indicates that labelling effort could be reduced by as much as 85%. Changing number of samples queried per active learning iteration offers achieving the same performance in lower number of iterations, but with the same labelling effort. Setting lower threshold for stream-based uncertainty acquisition function provides steeper increase in performance, while setting higher threshold offers a longer lasting active learning process.

Further work is required to develop a batch-aware and computationally efficient query strategy that is specifically designed for NILM. Also, interaction of NILM domain experts or end users would be worth exploring as well to determine the impact of labelling error, as people are prone to making errors when providing labels for queried samples, and their trust and confidence in using AI can be affected by the process.

Funding

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 955422.

CRediT authorship contribution statement

Tamara Todic: Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Visualization. **Vladimir Stankovic:** Conceptualization, Validation, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Lina Stankovic:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

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

An open-access, publicly available dataset was used in this study. The dataset can be accessed at https://pureportal.strath.ac.uk/en/datasets/refit-electrical-load-measurements-cleaned, hosted at University of Strathclyde [53].

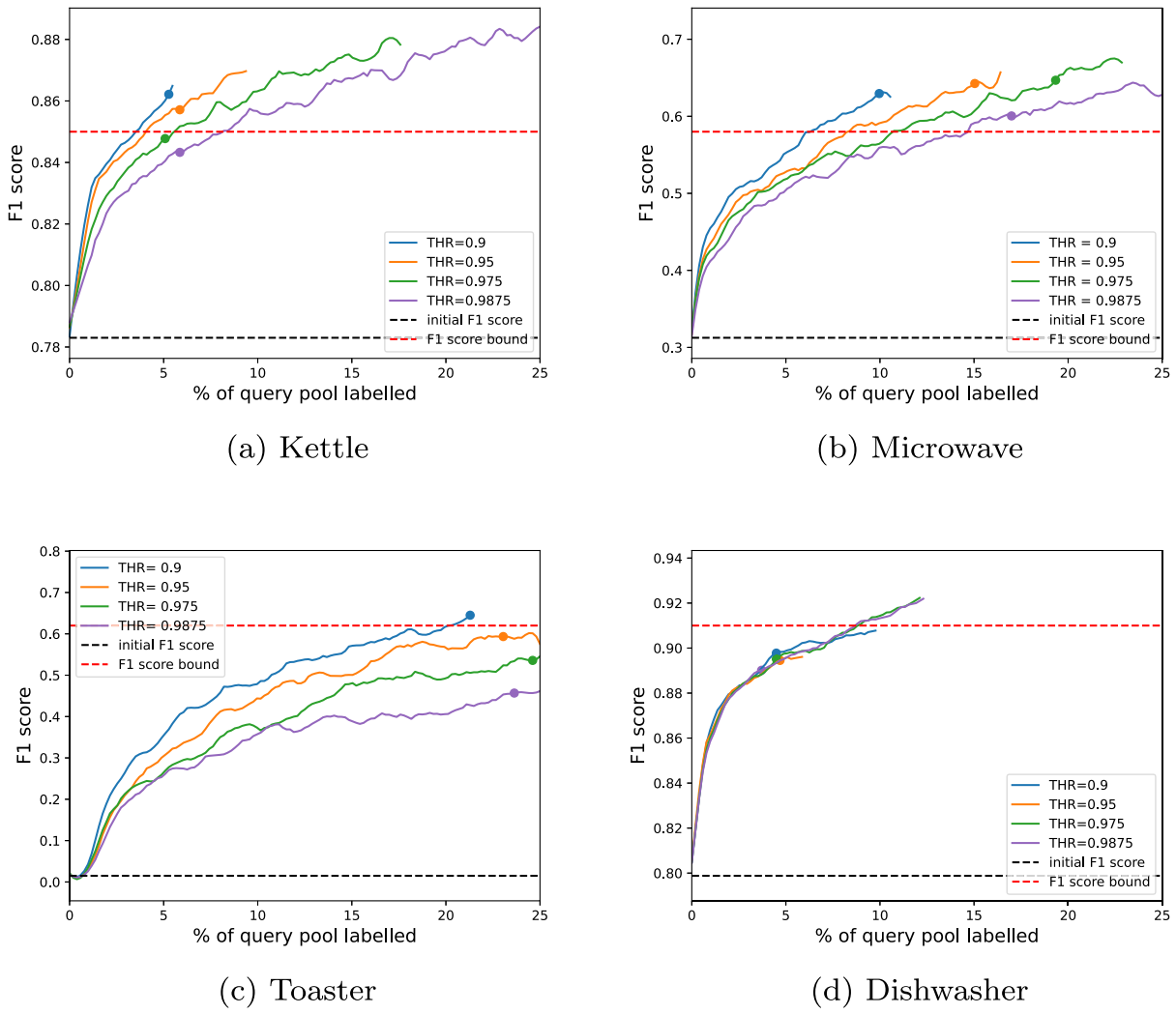


Fig. 8. Experiment 3 — sensitivity analysis: Models pre-trained with large datasets transferred to REFIT House 2 for kettle (a), microwave (b), toaster (c) and dishwasher (d). Fine-tuning of the model is performed in each active learning iteration using the stream-based uncertainty acquisition function with different confidence thresholds (THR).

References

[1] Delivering the european green deal. European Commission; 2022, https://ec.europa.eu/info/strategy/priorities-2019-2024/european-green-deal/delivering-european-green-deal_en [Accessed on 1 November 2022].

[2] Hart G. Nonintrusive appliance load monitoring. Proc IEEE 1992;80(12):1870–91. http://dx.doi.org/10.1109/5.192069.

[3] Stankovic L, Stankovic V, Liao J, Wilson C. Measuring the energy intensity of domestic activities from smart meter data. Appl Energy 2016;183:1565–80. http://dx.doi.org/10.1016/j.apenergy.2016.09.087.

[4] He K, Stankovic L, Liao J, Stankovic V. Non-intrusive load disaggregation using graph signal processing. IEEE Trans Smart Grid 2018;9(3):1739–47. http://dx.doi.org/10.1109/TSG.2016.2598872.

[5] Shi X, Ming H, Shakkottai S, Xie L, Yao J. Nonintrusive load monitoring in residential households with low-resolution data. Appl Energy 2019;252:113283. http://dx.doi.org/10.1016/j.apenergy.2019.05.086.

[6] Zhao B, He K, Stankovic L, Stankovic V. Improving event-based non-intrusive load monitoring using graph signal processing. IEEE Access 2018;6:53944–59.

[7] 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.

[8] Luan W, Wei Z, Liu B, Yu Y. Non-intrusive power waveform modeling and identification of air conditioning load. Appl Energy 2022;324:119755. http://dx.doi.org/10.1016/j.apenergy.2022.119755.

[9] He K, Jakovetic D, Zhao B, Stankovic V, Stankovic L, Cheng S. A generic optimisation-based approach for improving non-intrusive load monitoring. IEEE Trans Smart Grid 2019;10(6):6472–80.

[10] Himeur Y, Alsalemi A, Bensaali F, Amira A. Robust event-based non-intrusive appliance recognition using multi-scale wavelet packet tree and ensemble bagging tree. Appl Energy 2020;267:114877. http://dx.doi.org/10.1016/j.apenergy.2020.114877.

[11] Liao J, Elafoudi G, Stankovic L, Stankovic V. Non-intrusive appliance load monitoring using low-resolution smart meter data. In: 2014 IEEE international conference on smart grid communications. IEEE; 2014, p. 535–40.

[12] Mollel R, Stankovic L, Stankovic V. Using explainability tools to inform NILM algorithm performance: A decision tree approach. In: 6th international workshop on non-intrusive load monitoring. 2022, p. 1–5.

[13] Zhao B, Stankovic L, Stankovic V. On a training-less solution for non-intrusive appliance load monitoring using graph signal processing. IEEE Access 2016;4:1784–99.

[14] Altrabalsi H, Stankovic V, Liao J, Stankovic L. Low-complexity energy disaggregation using appliance load modelling. Aims Energy 2016;4(1):884–905.

[15] Himeur Y, Alsalemi A, Bensaali F, Amira A. Smart non-intrusive appliance identification using a novel local power histogramming descriptor with an improved k-nearest neighbors classifier. Sustainable Cities Soc 2021;67:102764. http://dx.doi.org/10.1016/j.scs.2021.102764.

[16] Angelis G-F, Timplalexis C, Krinidis S, Ioannidis D, Tzovaras D. NILM applications: Literature review of learning approaches, recent developments and challenges. Energy Build 2022;111951.

[17] Kim H, Marwah M, Arlitt M, Lyon G, Han J. Unsupervised disaggregation of low frequency power measurements. In: Proceedings of the 2011 SIAM international conference on data mining. SIAM; 2011, p. 747–58.

[18] Parson O, Ghosh S, Weal M, Rogers A. Non-intrusive load monitoring using prior models of general appliance types. In: Twenty-sixth AAAI conference on artificial intelligence. 2012.

[19] Kolter JZ, Jaakkola T. Approximate inference in additive factorial hmms with application to energy disaggregation. In: Artificial intelligence and statistics. PMLR; 2012, p. 1472–82.

[20] Zhong M, Goddard N, Sutton C. Signal aggregate constraints in additive factorial HMMs, with application to energy disaggregation. Adv Neural Inf Process Syst 2014;27.

[21] Liu C, Akintayo A, Jiang Z, Henze GP, Sarkar S. Multivariate exploration of non-intrusive load monitoring via spatiotemporal pattern network. Appl Energy 2018;211:1106–22. http://dx.doi.org/10.1016/j.apenergy.2017.12.026.

[22] 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.

[23] Settles B. Active learning literature survey. Computer sciences Technical report 1648, University of Wisconsin–Madison; 2009.

[24] Ren P, Xiao Y, Chang X, Huang P-Y, Li Z, Gupta BB, et al. A survey of deep active learning. ACM Comput Surv 2021;54(9):1–40.

[25] Wang W, Chen P, Xu Y, He Z. Active-MTSAD: Multivariate time series anomaly detection with active learning. In: 2022 52nd Annual IEEE/IFIP international conference on dependable systems and networks. IEEE; 2022, p. 263–74.

[26] Gu Q, Dai Q, Yu H, Ye R. Integrating multi-source transfer learning, active learning and metric learning paradigms for time series prediction. Appl Soft Comput 2021;109:107583.

[27] Jarl S, Aronsson L, Rahrovani S, Chehreghani MH. Active learning of driving scenario trajectories. Eng Appl Artif Intell 2022;113:104972.

[28] Jin X. Active learning framework for non-intrusive load monitoring. Tech. rep., Golden, CO (United States): National Renewable Energy Lab.(NREL); 2016.

[29] Liebgott F, Yang B. Active learning with cross-dataset validation in event-based non-intrusive load monitoring. In: 2017 25th european signal processing conference. IEEE; 2017, p. 296–300.

[30] Fatouh AM, Nasr OA, Eissa M. New semi-supervised and active learning combination technique for non-intrusive load monitoring. In: 2018 IEEE international conference on smart energy grid engineering. IEEE; 2018, p. 181–5.

[31] Guo L, Wang S, Chen H, Shi Q. A load identification method based on active deep learning and discrete wavelet transform. IEEE Access 2020;8:113932–42.

[32] Jiang J, Kong Q, Plumbley MD, Gilbert N, Hoogendoorn M, Roijers DM. Deep learning-based energy disaggregation and on/off detection of household appliances. ACM Trans Knowl Discov Data (TKDD) 2021;15(3):1–21.

[33] Huber P, Calatroni A, Rumsch A, Paice A. Review on deep neural networks applied to low-frequency nilm. Energies 2021;14(9):2390.

[34] do Nascimento PPM. Applications of deep learning techniques on NILM. Diss. Universidade Federal Do Rio de Janeiro; 2016.

[35] Zhang C, Zhong M, Wang Z, Goddard N, Sutton C. Sequence-to-point learning with neural networks for non-intrusive load monitoring. In: Proceedings of the AAAI conference on artificial intelligence, Vol. 32, no. 1. 2018.

[36] Rafiq H, Shi X, Zhang H, Li H, Ochani MK. A deep recurrent neural network for non-intrusive load monitoring based on multi-feature input space and post-processing. Energies 2020;13(9):2195.

[37] 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 international conference on acoustics, speech and signal processing. IEEE; 2019, p. 8330–4.

[38] 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.

[39] Li D, Li J, Zeng X, Stankovic V, Stankovic L, Xiao C, et al. Transfer learning for multi-objective non-intrusive load monitoring in smart buildings. Appl Energy 2022.

[40] Kirsch A, Van Amersfoort J, Gal Y. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. Adv Neural Inf Process Syst 2019;32.

[41] Li H, Wang Y, Li Y, Xiao G, Hu P, Zhao R. Batch mode active learning via adaptive criteria weights. Appl Intell 2021;51:3475–89.

[42] Zhao B, Ye M, Stankovic L, Stankovic V. Non-intrusive load disaggregation solutions for very low-rate smart meter data. Appl Energy 2020;268:114949. http://dx.doi.org/10.1016/j.apenergy.2020.114949.

[43] Zeifman M, Roth K. Nonintrusive appliance load monitoring: Review and outlook. IEEE Trans Consum Electron 2011;57(1):76–84. http://dx.doi.org/10.1109/TCE.2011.5735484.

[44] Zoha A, Gluhak A, Imran MA, Rajasegarar S. Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey. Sensors 2012;12(12):16838–66.

[45] Batra N, Kukunuri R, Pandey A, Malakar R, Kumar R, Krystalakos O, et al. Towards reproducible state-of-the-art energy disaggregation. In: Proceedings of the 6th ACM international conference on systems for energy-efficient buildings, cities, and transportation. 2019, p. 193–202.

[46] Reinhardt A, Klemenjak C. How does load disaggregation performance depend on data characteristics? insights from a benchmarking study. In: Proceedings of the eleventh ACM international conference on future energy systems. 2020, p. 167–77.

[47] Pan Y, Liu K, Shen Z, Cai X, Jia Z. Sequence-to-subsequence learning with conditional gan for power disaggregation. In: ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing. IEEE; 2020, p. 3202–6.

[48] 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.

[49] Massidda L, Marrocu M, Manca S. Non-intrusive load disaggregation by convolutional neural network and multilabel classification. Appl Sci 2020;10(4):1454.

[50] 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.

[51] Wang L, Mao S, Nelms RM. Transformer for non-intrusive load monitoring: Complexity reduction and transferability. IEEE Internet Things J 2022.

[52] Kolter JZ, Johnson MJ. REDD: A public data set for energy disaggregation research. In: Workshop on data mining applications in sustainability, Vol. 25. San Diego, CA: Citeseer; 2011, p. 59–62.

[53] Murray D, Stankovic L, Stankovic V. An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study. Sci Data 2017;4(1):1–12.

[54] 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.

[55] Wang L, Mao S, Wilamowski BM, Nelms RM. Pre-trained models for non-intrusive appliance load monitoring. IEEE Trans Green Commun Netw 2021;6(1):56–68.

[56] D'Incecco M, Squartini S, Zhong M. Transfer learning for non-intrusive load monitoring. IEEE Trans Smart Grid 2019;11(2):1419–29.

[57] Lüth CT, Bungert TJ, Klein L, Jaeger PF. Toward realistic evaluation of deep active learning algorithms in image classification. 2023, arXiv preprint arXiv:2301.10625.

[58] Li Z, Zhao Y, Geng Y, Zhao Z, Wang H, Chen W, et al. Situation-aware multivariate time series anomaly detection through active learning and contrast VAE-based models in large distributed systems. IEEE J Sel Areas Commun 2022;40(9):2746–65.

[59] Filip A, et al. Blued: A fully labeled public dataset for event-based nonintrusive load monitoring research. In: 2nd workshop on data mining applications in sustainability, Vol. 2012. 2011.

[60] Gao J, Giri S, Kara EC, Bergés M. Plaid: A public dataset of high-resoultion electrical appliance measurements for load identification research: Demo abstract. In: Proceedings of the 1st ACM conference on embedded systems for energy-efficient buildings. 2014, p. 198–9.

[61] Kahl M, Haq AU, Kriechbaumer T, Jacobsen H-A. Whited-a worldwide household and industry transient energy data set. In: 3rd international workshop on non-intrusive load monitoring. 2016, p. 1–4.

[62] Picon T, Meziane MN, Ravier P, Lamarque G, Novello C, Bunetel J-CL, et al. COOLL: Controlled on/off loads library, a public dataset of high-sampled electrical signals for appliance identification. 2016, arXiv preprint arXiv:1611.05803.

[63] Gal Y, Ghahramani Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: Int. conf. machine learning. PMLR; 2016, p. 1050–9.

[64] Brust C-A, Käding C, Denzler J. Active and incremental learning with weak supervision. KI-KÜNstliche Intell 2020;34:165–80.