Intelligent Data Analysis for Energy Management

XIAOMIN CHANG

Doctor of Philosophy

THE UNIVERSITY OF
SYDNEY

Supervisor: Albert Zomaya
Associate Supervisor: Wei Li

A thesis submitted in fulfilment of
the requirements for the degree of
Doctor of Philosophy

School of Computer Science
Faculty of Engineering
The University of Sydney
Australia

13 July 2023

Publications

[1] Xiaomin Chang, Wei Li, Jin Ma, Ting Yang, and Albert Y. Zomaya. "Interpretable machine learning in sustainable edge computing: A case study of short-term photovoltaic power output prediction." In Proceedings of the 45th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2020

[2] Xiaomin Chang, Wei Li, and Albert Y. Zomaya. "A lightweight short-term photovoltaic power prediction for edge computing." IEEE Transactions on Green Communications and Networking (TGCN), 2020

[3] Xiaomin Chang, Wei Li, Chunqiu Xia, Qiang Yang, Jin Ma, Ting Yang, and Albert Zomaya. "Transferable Tree-based Ensemble Model for Non-Intrusive Load Monitoring." IEEE Transactions on Sustainable Computing (T-SUSC), 2022

[4] Xiaomin Chang, Wei Li, Yunchuan Shi and Albert Y. Zomaya. "Taming the Domain Shift in Multi-source Learning for Energy Disaggregation." In Proceedings of the 29th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD), 2023

[5] Xiaomin Chang, Wei Li, and Albert Y. Zomaya. "Fed-GBM: a cost-effective federated gradient boosting tree for non-intrusive load monitoring." In Proceedings of the 13th ACM International Conference on Future Energy Systems (e-Energy), 2022

[6] Yunchuan Shi, Wei Li, Xiaomin Chang, Ting Yang, Yaojie Sun, and Albert Y. Zomaya. "On Enabling Collaborative Non-Intrusive Load Monitoring for Sustainable Smart Cities." Scientific reports, 2023

[7] Yunchuan Shi, Wei Li, Xiaomin Chang, and Albert Y. Zomaya. "User privacy leakages from federated learning in NILM applications." In Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys), 2021

[8] Chunqiu Xia, Wei Li, Xiaomin Chang, Tianming Zhao, and Albert Zomaya. "Lightweight Online Scheduling for Home Energy Management Systems under Uncertainty." IEEE Transactions on Sustainable Computing (T-SUSC), 2022

[9] Zezheng Zhao, Chunqiu Xia, Lian Chi, Xiaomin Chang, Wei Li, Ting Yang, and Albert Y. Zomaya. "Short-Term Load Forecasting Based on the Transformer Model." Information, 2021

[10] Zezheng Zhao, Chunqiu Xia, Lian Chi, Xiaomin Chang, Wei Li, Ting Yang, and Albert Y. Zomaya. "An Adaptive Multi-objective Salp Swarm Algorithm for Efficient Demand Side Management." In Proceedings of the 17th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 2020

[11] Chunqiu Xia, Wei Li, Xiaomin Chang, and Albert Y. Zomaya. "Online Energy Management under Uncertainty for Net-Zero Energy Ecosystems." In Proceedings of the International Conference on Internet-of-Things Design and Implementation (IoTDI), 2021

[12] Jiuling Li, Wei Li, Xiaomin Chang, Keshab Sharma, and Zhiguo Yuan. "Real-time predictive control for chemical distribution in sewer networks using improved elephant herding optimization." IEEE Transactions on Industrial Informatics, 2020

[13] Chunqiu Xia, Wei Li, Xiaomin Chang, Ting Yang, and Albert Y. Zomaya. "A rank-based multiple-choice secretary algorithm for minimising microgrid operating cost under uncertainties." Frontiers in Energy (2023)

Authorship Attribution Statement

Chapter 2 of this thesis is published as [1] and [2]. I performed the majority of the research, analyzed the data, and wrote the drafts of the paper.

Chapter 3 of this thesis is published as [3]. I performed the majority of the research, analyzed the data, and wrote the drafts of the paper.

Chapter 4 of this thesis is published as [4]. I performed the majority of the research, analyzed the data, and wrote the drafts of the paper.

Chapter 5 of this thesis is published as [5]. I performed the majority of the research, analyzed the data, and wrote the drafts of the paper.

I certify that the aforementioned authorship attribution statements are correct and I have received permission from the other authors to include the published materials.

Name: Xiaomin Chang

Signature: **Date**:

As the supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Supervisor Name: Albert Y. Zomaya

Signature: **Date**:

Statement of Originality

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes.

I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Name: XIAOMIN CHANG

Signature: **Date**:

Student Plagiarism: Compliance Statement

I certify that:

I have read and understood the University of Sydney Student Plagiarism: Coursework Policy and Procedure;

I understand that failure to comply with the Student Plagiarism: Coursework Policy and Procedure can lead to the University commencing proceedings against me for potential student misconduct under Chapter 8 of the University of Sydney By-Law 1999 (as amended);

This Work is substantially my own, and to the extent that any part of this Work is not my own I have indicated that it is not my own by Acknowledging the Source of that part or those parts of the Work.

Name: XIAOMIN CHANG

Signature: **Date**:

Abstract

Predictive data analysis has been identified as essential to support intelligent energy management for better energy sustainability and efficiency. Previous studies have showcased that predicted energy information can benefit consumers economically by optimising energy usage while assisting energy suppliers in efficiently planning power distribution and implementing DR energy management. Recent advances in the Internet of Things (IoT) and Information and Communication Technologies (ICT) simplify the collection of desired energy data streams for further informatics analysis. With such energy data, machine learning (ML) prevails to effectively infer future knowledge associated with online energy resource scheduling, e.g., renewable energy generation, load demands and electricity prices. Although some early efforts have been dedicated to incorporating ML into energy management, computation resource limitations and data scarcity are two pressing challenges for on-site predictive energy analysis. Due to privacy concerns, users prefer on-premise model establishment instead of placing the training task in the cloud and sharing sensitive energy data. But most ML algorithms rely heavily on solid computational resources and vast amounts of labelled data to succeed. Users are often unable to fulfil the requirements in real-world scenarios. To this end, this thesis uses different perspectives to propose several affordable solutions for performing on-demand intelligent data analysis on local resource-constrained devices. Also, three algorithm-specific training frameworks have been developed to solve data shortage by leveraging easily obtainable but extensive data sources based on transfer learning and federated learning. We implement our design under practical settings for photovoltaic (PV) power prediction and non-intrusive load monitoring (NILM) as case studies to fully evaluate their performances.

Acknowledgements

This thesis is the culmination of several years of research work to pursue a Doctor of Philosophy at the University of Sydney. During such an unforgettable journey, countless people made invaluable contributions to the production of this thesis. I would like to express my sincere gratitude to everybody.

My first and big appreciation goes to my principal supervisor, Prof. Albert Y. Zomaya, at the University of Sydney, for his marvellous supervision, guidance, and encouragement throughout my master's and doctoral degrees. You are the true definition of a leader and the ultimate role model. I'm extremely grateful for providing me with an extraordinary research environment with sufficient resources, which enables me to grow professionally. Your productive and insightful suggestions had a major impact on my thesis. It was a great privilege and honour to work and study under your guidance.

Also, I would like to thank my co-supervisor, Wei Li at the University of Sydney, who has been an unceasing source of advice, insight, and good judgment. Your patience, motivation, and enthusiasm have deeply inspired me through my research. Thanks also for your time in reading and correcting my manuscripts, even though you were overwhelmed with work. I have significantly benefited from your wealth of knowledge and meticulous editing. Whenever I ran into a trouble spot or had a question about my research, your marvellous guidance and encouragement always steered me in the right direction.

I gratefully recognize the invaluable assistance of the experts: Ting Yang, Jin Ma, and Qiang Yang. Without your passionate participation and input, I could not have effectively completed all my research work and papers. I would like to thank my lab mates – Chunqiu Xia, Shaojun Zhang, Yucen Nan, Zezheng Zhao, Xinwei Ji, Tianming Zhao and Yunchuan Shi for your support in the smooth running of the lab and a cherished time spent together.

For financial support, I would particularly like to thank the University of Sydney International Scholarship (USydIS) funded by the University of Sydney. The scholarship supports to relieve my financial burden, guaranteeing the high quality of my research.

I am especially grateful to my parents for their endless support. You have always stood behind me, and this was no exception. With your love, caring, and financial support, I had the strength to pursue my dream and choose what I desired.

Last but not least, heartfelt thanks go to my caring, loving, and supportive wife, Qian Wu. Your encouragement when the times got rough is much appreciated and duly noted. It was a great comfort and relief to know that you were willing to provide management of our household activities while I completed my work.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Energy consumption has increased dramatically due to continued rapid economic growth, the industrial revival, and the diverse life-related human demands. Energy resource exhaustion and negative environmental footprint have raised significant concerns worldwide and become critical areas of interest in different research communities. From [46, 20], it is worth noting that electricity generation comprises 25% of the world's carbon emissions, and up to 41% of major energy consumption comes from the residential, commercial, and public service sectors. In past decades, a considerable number of studies have focused on improving energy efficiency and environmental sustainability in these sectors [117, 132, 48]. The concept of sustainable computing has flourished in the Information Technology (IT) field. The wide deployment of modern computing systems brings imperative demands for solving energy-related issues. The main direction for such a goal is to explore energy schedulers, which can orchestrate infrastructures to perform intelligent control of energy usage based on multi-source energy generation.

As the most popular scheme for energy management, demand response (DR) has been widely adopted to reduce the end users' electricity tariff and increase the utilisation of renewable energies by flattening the load profile according to dynamic electricity prices [6]. DR also helps the energy supplier relieve the high pressure of electricity distribution and lower the risk of blackouts due to overload around peak hours, improving the stability of the power grid. Previous studies have proposed substantial energy management policies based on the DR scheme from different perspectives, facilitating the development of energy management systems (EMS) [24, 138]. However, uncertainties and intermittency, which inherently exist in load demands and renewable energy generation, pose significant challenges to the online

optimisation of energy usage. Emerging energy storage techniques further complicate real-time decision making for EMS. As such, addressing such uncertainties is a prerequisite for successful energy management. Fortunately, the advent of Machine learning (ML) techniques empowers predictive information analysis of load demands and renewable energy generation, making them essential tools for alleviating negative impacts due to uncertainties.

Energy usage and generation information has been recognised as key to improving energy efficiency and environmental sustainability. Recent advances in the Internet of Things (IoT) and Information and Communication Technologies (ICT) make it easy to achieve information exchange and collect desired energy data for further informatics analysis [102, 94]. With sufficient appliance-level consumption data, powerful ML approaches enable deep mining of consumer usage patterns, reflecting various indoor activities practiced in daily life. Also, various tailored ML algorithms allow practical forecasts on renewable energy generation based on historical records, though such generation correlates to multiple environmental variables with robust randomness, such as irradiance, temperature, humidity, and wind speed. Consumers can reap economic benefits by integrating sustainable energy and optimising energy management using predicted energy information. Such information can help the energy suppliers efficiently schedule power distribution and implement DR, whilst also considering consumers' comfort.

However, the success of most ML algorithms relies highly on powerful computation resources and large-scale labelled data [136]. In real-life scenarios, users are not always able to meet the requirements. As we know, energy data is embedded with sensitive information about individuals. Due to privacy concerns, electricity users prefer on-site model training, rather than placing the training work in a cloud center. The resource-constrained local devices are incapable of handling substantial workloads with diverse computational patterns, such as rapid gradient computation, complex matrix calculation, and large-scale optimum search. To this end, we propose a cost-effective means to perform on-demand intelligent data analysis at local places using edge devices. As a case study, we implement photovoltaic (PV) power prediction to thoroughly demonstrate the outstanding performance of our solution. Collecting and labelling energy data is prohibitively labour-expensive, time-consuming, and error-prone.

Furthermore, massive sensing equipment significantly increases capital investment at user premises. The lack of labelled data will lead to failure in model construction for the target task. In practice, a sufficient amount of data sources from other domains are available for model training. Thus, effectively leveraging accessible data sources to train a model safely is a core research challenge. Federated learning and transfer learning techniques are vital solutions to data scarcity. They also bring fruitful benefits to other predictive analysis applications. Prior works utilising these two techniques, however, are limited in the energy field. In this thesis, we fill the technical gaps and propose three algorithm-specific training methods to solve data shortage. Two correspond to transfer learning, and one is federated learning-based. In this thesis, we focus on the energy disaggregation task and examine the performances of our solution by implementing these three methods under suitable settings according to human demands and environmental constraints.

To be specific, this research makes the following contributions:

- To better leverage solar energy, we provide a lightweight and interpretable analytic solution for reducing the limitations of evolving PV systems. In this work, we create a cost-effective clustering-based training framework for achieving accurate PV output forecasting. Our framework orchestrates different predictors to lower forecast errors by combining weather clustering and temporal pattern aggregation. Bayesian optimisation is applied to find the appropriate time step size to learn full temporal meteorological trends, whilst reaching an optimal trade-off between accuracy and running costs. We first attempt to integrate two variants of gradient boosting decision trees (GBDT) into our designed framework due to their low computational cost and impressive performance for non-linear regression problems. To investigate our solution's performance, we implement it on a single powerful machine, then on a resource-constrained edge cluster, and conduct three groups of comparative experiments with other benchmark algorithms. The results show that our approach outperforms the other learning algorithms and demonstrates the potential of boosting trees for informative on-site prediction, especially when only edge devices with limited computation capacities are available.

- We also extend the application of GBDT to non-intrusive load monitoring (NILM) and tackle two practical issues, data scarcity and domain shift. From our previous work, LightGBM, as an effective implementation of GBDT, exhibits superiority in accuracy and cost-efficiency compared with other ML algorithms. Hence, we choose to use it as a base model to achieve energy disaggregation. We stitch together the sequence-to-point training paradigm and LightGBM to thoroughly learn the signatures of appliances from a long-term time series with low computation and storage overhead. However, data shortage raises major concerns in practical applications since data collection and manual annotation are time-consuming and expensive. Lack of diversified training data often leads to low generality of boosting trees when serving cross-domain tasks, due to statistical misalignment between different domains. Also, the establishment of a new model, associated with considerable hyper-parameter updates, will incur high computation overhead, even if limited valuable target data can be used for training. To address such limitations, we propose an adaptive model-based transfer learning approach for GBDT, enabling the re-use of the knowledge in a fully-trained model. As the source data is privacy sensitive for end users, our solution is designed for source-free training environments where only limited labelled target data can be accessed. The experiment results indicate that our developed approach successfully realises model transfer and maintains comparable accuracy in the unseen target domain.
- We address data shortage and domain shift issues encountered by deep networks for NILM applications. Currently, massive open-source datasets are available for NILM studies, offering opportunities to improve the generality of prediction models. This research trend urges a new need for a feasible method to realise multi-source learning and alleviate the impacts of domain shift. In this work, we mathematically formulate the multi-source domain adaptation for energy disaggregation and prove a new generalisation bound of target risk in terms of weighted source risk and domain discrepancy. Our induced domain discrepancy jointly considers the marginal shift and conditional shift. Based on our bound, we develop a hybrid loss-driven multi-source domain adversarial network (HLD-MDAN) to learn domain-invariant feature

representations and structures for cross-domain tasks. Extensive experimental studies validate our solution's efficacy and superiority in single-source and multi-source learning settings, compared with other state-of-the-art algorithms.

- We propose the use of a cost-effective collaborative learning framework, namely Fed-GBM, to overcome the low model fitting of local on-site training in NILM applications due to restricted data size. In Fed-GBM, a two-stage voting scheme is employed to protect the users from privacy leakage and reduce communication costs by locally filtering out invaluable data (histograms in the tree growth). To boost resource utilisation, node parallelism for tree development is also introduced with an online scheduling technique. Such technical designs greatly lower training costs across different participants. We conduct extensive experimental research on Fed-GBM, with the results proving that it performs admirably in all benchmark assessments. This work broadens the horizon of GBDT algorithms for informative analysis in the energy field and makes it possible to handle complex environments.

The remaining parts of the thesis are organised as follows. Chapter 2 introduces a lightweight cluster-based training framework to overcome the limitations of PV systems and resource constraints for edge learning. It also examines the feasibility and cost-efficiency of GBDT for time-series predictive analysis. Chapter 3 extends the learning method to the NILM problem. A source-free model-based transfer learning approach is proposed to mitigate the effects of data scarcity in GBDT modelling. Chapter 4 delves deeper into the challenges of data shortage and domain shift encountered by deep neural networks in NILM applications. It introduces a novel multi-source learning framework and a domain adaptation approach for cross-domain tasks. Unlike the application scenario in Chapter 3, where source data privacy is a concern, this approach is particularly suitable when sufficient labelled source data are open-source and easy-to-get for model training. Building upon the insights gained from Chapter 2 and 3, Chapter 5 presents an alternative approach to address data scarcity from the collaborative aspect. It proposes a cost-effective horizontal federated learning framework, Fed-GBM, to enable the implementation of joint model training across different end users. Chapter 6 presents the conclusion of all research works and briefly discusses future work.

CHAPTER 2

Lightweight Photovoltaic Power Prediction for Edge Computing

To meet the need for energy savings in Internet of Things (IoT) systems, solar energy has been increasingly exploited to serve as a green and renewable source to allow systems to better operate in an energy-efficient way. In this respect, accurate PV power output prediction is a prerequisite for any energy-saving scheme employed in these systems. In this work, we propose a unified training framework combined with the tree-boosting algorithm to obtain a prediction model, which can provide short-term predictions of PV power output. Compared with the training in a single powerful machine, our proposed framework is more energy-efficient and fits into devices with limited computation and storage resources. The experimental results show that our proposed framework is superior to other benchmark machine learning algorithms.

2.1 Introduction

In recent years, the Internet of Things (IoT) paradigm has been increasingly applied to all spheres of our daily lives. IoT applications can be found in a wide range of daily activities, including home automation, health care, agriculture, smart cities and many more; and as an enabler for bidirectional object-object and human-object interactions, eventually realising the vision of a fully connected world. As reported in [106], it was expected that there would be 20.4 billion IoT devices deployed worldwide by 2020. With the dense deployment of IoT devices, IoT applications will inevitably generate a massive amount of data, which has to be processed, stored and properly accessed by end users [61]. Edge computing [18], as an emerging distributed computing paradigm, has been introduced to enable data processing

and on-demand services at the edge of the network for IoT applications. Thus, the desired capabilities of large scale cloud data centers, such as rich resources of computation and storage, can be used by IoT devices from the edge of the network.

Considering the growing concerns over building greener and more sustainable computing systems, in [59], the authors argue that energy related issues are equally important to other research efforts of edge computing, e.g. flexibility, scalability, security, programmability, and real-time processing. To help reduce the energy consumption of edge computing systems, the use of renewable energy has been considered an effective and sustainable means to achieve the objective. To date, the most widely used and reliable renewable energy resource is solar energy where electricity can be easily produced from photovoltaic (PV) panels. However, solar power generation is heavily reliant on multiple meteorological factors, e.g. solar irradiance, cloud opacity, and air temperature. Considering the uncertain nature of meteorological factors, accurate PV power output prediction plays a vital role to enable the smooth running of edge computing systems with solar energy supply. A number of research studies have been dedicated to developing prediction models with the focus on either improving forecasting accuracy or lowering the computation overhead. The existing approaches used for PV output prediction can be broadly classified into two types [30]: classic statistical algorithms, including numerical weather prediction (NWP) and Auto-Regressive and Moving Average Model (ARMA); and machine learning algorithms, including extreme learning machine (ELM), support vector regression (SVR), general regression neural network (GRNN) and Artificial Neural Network (ANN). However, the models that can provide high accuracy in prediction mostly need high computation and storage capacities at run-time. They generally perform poorly in edge computing systems with limited resource. On the other hand, lightweight algorithms are unable to provide good prediction results for further usage.

To tackle this issue, in this work, we develop an effective clustering-based training framework as a computational model for predicting PV power output. In the framework, we integrate temporal pattern aggregation with weather clustering to proactively reduce prediction errors. To comprehensively consider temporal meteorological patterns, a Bayesian optimisation algorithm is employed to search for the optimal time step size which leads to minimum

prediction errors. In addition to that, we utilise two tree-based machine learning methods, tree-structured self-organising map (TS-SOM) and tree-gradient boosting algorithm XGBoost/LightGBM, to realise the weather clustering and the training of the regression model respectively. By using our proposed approach, accurate PV power output prediction can be achieved with low computation overhead which makes it suitable for use on a wide range of edge devices. The main contributions of this chapter are:

(1) We develop a lightweight computational framework for PV output prediction, which significantly reduces the prediction error by explicitly utilising the temporal sequential patterns of solar energy generation.
(2) We explore the correlations between the time horizon of selected meteorological patterns and prediction accuracy and integrate Bayesian optimisation algorithms into our designed training framework to search for the optimal time steps for temporal pattern aggregation.
(3) We implement two variants of boosting trees on an edge computing platform with limited computation and storage resources to achieve better system energy efficiency.

2.2 Background

In this section, we investigate state-of-the-art techniques for PV output prediction and discuss their respective features and contributions. Conventionally, there are a number of statistical learning algorithms devoted to performing PV output prediction, such as Auto-Regressive and Moving Average Model (ARMA) and Autoregressive Integrated Moving Average Model (ARIMA) [96]. These methods enable direct prediction based on historical power output data collected from PV monitor systems and exhibit time series characteristics of solar energy generation. However, these approaches cannot comprehensively consider all the physical factors for PV output prediction, inevitably resulting in high deviation between targets and predictions. By contrast, a number of indirect prediction methods have achieved increasing popularity in recent research, which select diverse meteorological patterns as input variables, such as solar radiation, temperature, wind speed, wind direction, and humidity [57]. It has

been proved that there is a strong correlation between solar energy generation and some meteorological factors [133]. Thus, these indirect prediction approaches are broadly adopted and show good performance on accuracy in practice. These indirect prediction approaches can be further classified into traditional machine learning methods and artificial neural network (ANN) techniques.

Instead of performing complex analysis on meteorological information and power outputs to find specific rules for solar energy generation, the machine learning methods simplify the prediction process by solely learning model parameters based on historical data, and retaining the learned knowledge in a model. Support vector regression (SVR) is an extension of support vector machine (SVM) to address non-linear regression on solar energy generation forecast [32]. The computational complexity of SVR is not proportional to the dimensionality of the input space and SVR shows excellent generalization capability. However, it is challenging for SVR to perform training on large-scale sample data. Generalised regression neural network (GRNN) is also adopted for PV output prediction in [14]. The network of GRNN is trained by employing a single-pass learning strategy, thus it speeds up model training significantly. In addition, some other machine learning algorithms with respective advantages and drawbacks, such as K-nearest neighbours (KNN) [127], and multivariable linear regression (MLR) [108] have been applied in this field as well. Regarding ANN models, recent research increasingly uses ANN as the main prediction model to achieve solar energy generation forecasts. In particular, recurrent neural network (RNN) is developed to enable the capability of coping with temporal sequence problems and deliver excellent performance on nonlinear mapping. However, the accuracy of ANN closely depends on massive data and the selection of appropriate parameters [2].

In recent years, a number of ensemble methods have been explored to improve performance in this field, some of which provide better accuracy than a single prediction model. In [66], dynamic ensembles of neural networks are designed for one day-ahead PV output prediction. The work in [122] integrates three methods, including ELM, GRNN and Elman, to achieve short-term PV output prediction. In addition, tree-based ensemble methods, such as gradient boosting decision tree (GBDT) and XGBoost exploit their advantages and outperform most

TABLE 2.1: Weather features

Field	Description	Pearson correlation
ghi	Global horizontal irradiance for center value	0.9767
ghi10	Global horizontal irradiance for 10% value	0.9629
ghi90	Global horizontal irradiance for 90% value	0.9744
dni	Direct normal irradiance for center value	0.9275
dni10	Direct normal irradiance for 10% value	0.9077
dni90	Direct normal irradiance for 90% value	0.8644
dhi	Diffuse horizontal irradiance	0.9314
ebh	Direct horizontal irradiance	0.6617
air_temp	Air temperature	0.3350
zenith	Solar zenith angle. Range: 0∼180	-0.8013
azimuth	Solar azimuth angle. Range: -180∼180	-0.0092
cloud opacity	The quantity of cloud	-0.2184

single models on non-continuous prediction for short-term PV power output [89]. However, these approaches are time-consuming in model training, resulting in high memory usage and computation cost, which is not appropriate for edge devices. To address the above limitations, in this work, we leverage the power of the ensemble learning approach GBDT and combine it with our proposed cluster-based training framework to realise PV output prediction. Our solution can enhance prediction accuracy and lower computation and storage costs, which is suitable for edge computing environments with resource constraints and real-time prediction requirements.

2.3 Lightweight PV Prediction Model

In this section, we propose a lightweight computational framework for PV output prediction; its overall design is given in Figure 2.1. The framework is comprised of three functional components, namely temporal pattern aggregation, weather clustering, and model training. Also, the weather features shown in Table 2.1 are used to establish prediction models.

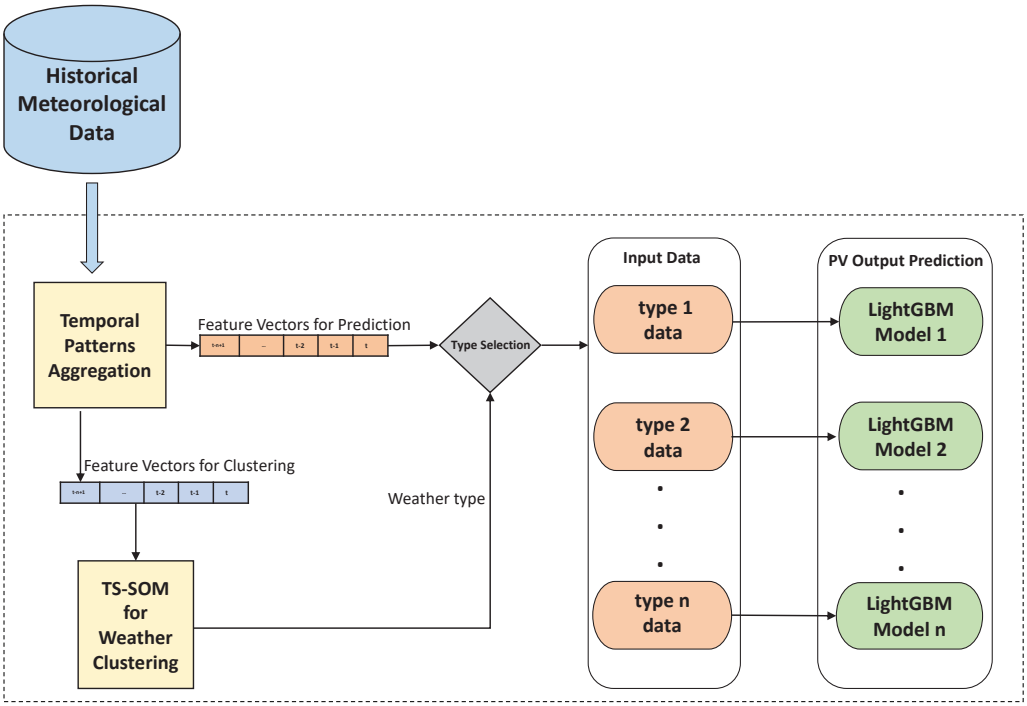


FIGURE 2.1: Overview of the framework for establishing prediction model

2.3.1 Temporal Patterns Aggregation

It is evident that meteorological factors can affect the PV power output, thus any noticeable changes in weather conditions will cause sudden spikes and falls to occur in the PV output curves [75]. If the temporal sequence pattern of the meteorological factors is not consciously considered, an inaccurate prediction could easily occur [121]. Thus, to further enhance the prediction accuracy, it is crucial to aggregate the temporal meteorological patterns before performing the weather clustering and training of the prediction model.

In this step, the iterative strategy as shown in Figure 2.2 is used to construct the dataset for weather clustering and model training respectively, but with different features. To construct the dataset for the use of weather clustering, we employed all features listed in Table 2.1 to ensure the factors affecting weather are maximally involved. Let W_t denote a vector of selected weather features at time t, and let n denote the selected size of time steps as the time horizon for the vector combination. At any time point t, all the feature vectors, ranging from

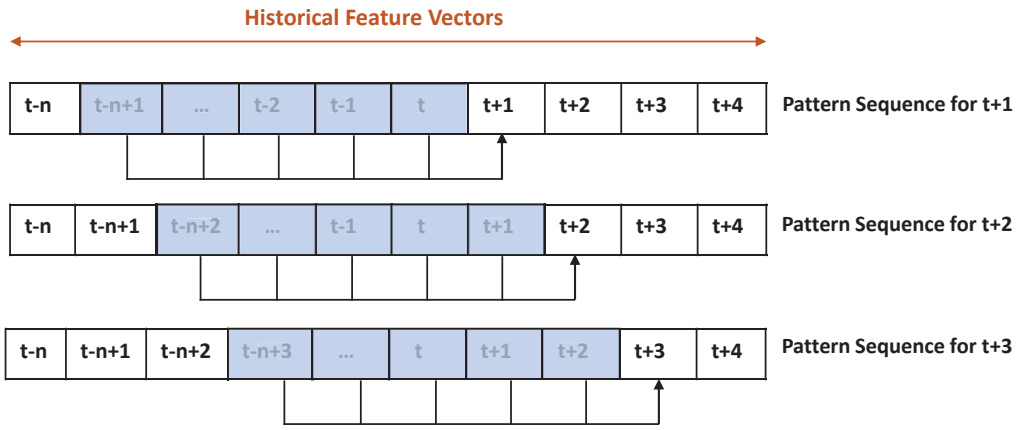


FIGURE 2.2: The iterative strategy for temporal pattern aggregation

t-n+1 to t, are combined as a new vector $C_{t+1} = \{W_{t-n+1}, ..., W_{t-1}, W_t\}$. The vector C_{t+1} represents the weather condition at time t+1 and will be used as a data sample for weather clustering. By doing so, new data samples at different future time points can be iteratively generated.

To determine the degree of correlation between the weather features and PV power output, Pearson correlation coefficients are used to establish the extent to which these features impact solar power generation. As listed in Table 2.1, it is not hard to observe that Global Horizontal Irradiance (GHI), Direct Normal Irradiance (DNI) and Diffuse Horizontal Irradiance (DHI) have a strong correlation with PV power output and Pearson correlation of these features is larger than 0.8. Besides, [84] demonstrates that the ratio of DHI and GHI can adequately reflect the dynamics of solar energy generation. Thus, we only select the weather features that represent solar irradiance as inputs to perform PV power output prediction model training, including GHI, DNI, DHI, and zenith, which is denoted as I. Besides, a certain range of the historical PV power output p is also used as input variables to reduce the computational cost by learning the historical pattern from the missing weather features. By employing the same strategy, at any time point t we aggregate all the selected feature vectors from t-n+1 to t as a new vector, which can be denoted as $F_{t+1} = \{I_{t-n+1}, p_{t-n+1}, ..., I_{t-1}, p_{t-1}, I_t, p_t\}$. The newly generated feature vector F_{t+1} considers the temporal sequence nature of the weather and PV power output, and enables comprehensive representation for PV power output at time t+1.

2.3.2 Searching for Optimal Time Steps

Algorithm 1 Searching for optimal time steps

Input: D is the observed meteorological data
1: T_{max}: is the maximum number of searching iteration
2: f: is the loss function for prediction model
3: ξ: is the search space
4:
Output: x_{best} is the selected time-step size resulting in lowest prediction errors
5: $S \leftarrow$ InitialSampling(D, f)
6: **for** t = T_0 to T_{max} **do**
7: p($Y|S$, D) $\leftarrow$ GaussBuild(S)
8: x$_{new}$ $\leftarrow$ argmax$_{x \in \xi}$ Acquisition(x, p($Y|S$, D))
9: $M \leftarrow$ BuildModel(x$_{new}$)
10: y$_{new}$ $\leftarrow$ f(M, D)
11: $S \leftarrow S \cup (x_{new}, y_{new})$
12: **end forreturn** x_{best}

To obtain the optimal size of time steps for aggregating temporal patterns, we employed a Bayesian optimization algorithm [97] in this work. A Bayesian optimization algorithm is a sequential search technique that enables a search for a global optimization solution for black-box functions in a cost-efficient way. Algorithm 1 is the pseudo code of using a Bayesian optimization algorithm to search for the optimal value. Initially, several sizes of time steps are randomly sampled and prediction errors for each selected time step size are calculated respectively. As depicted in Algorithm 1, the result S is a set of vectors (x_i, y_i). x_i denotes the i^{th} sampled size of time steps and y_i is the corresponding prediction loss for the model that is trained when x_i is selected as the size of time steps to achieve pattern aggregation. From line 2 to 8, a search for the optimal solution is performed iteratively. Within the iteration, the first step is to build a Gaussian Distribution Model. Based on the sample data contained in S, the values of mean and covariance kernel can be easily derived. Then, based on the Gaussian Distribution Model, we utilize a standard acquisition function, Expected Improvement Criterion (EI), to calculate the size of step times x_{new} that brings about maximum (EI) value. Once the x_{new} is selected, a new model for PV output prediction will be established and corresponding prediction loss can be calculated. Finally, new vector (x_{new}, y_{new}) is added into S. With this method, the optimal size of time steps can be obtained effectively and efficiently. Apart from Bayesian optimization, swarm intelligence

Optimization [15], genetic algorithm [44], or differential evolution [65] also can be taken into consideration for time step selection.

2.3.3 Weather Clustering

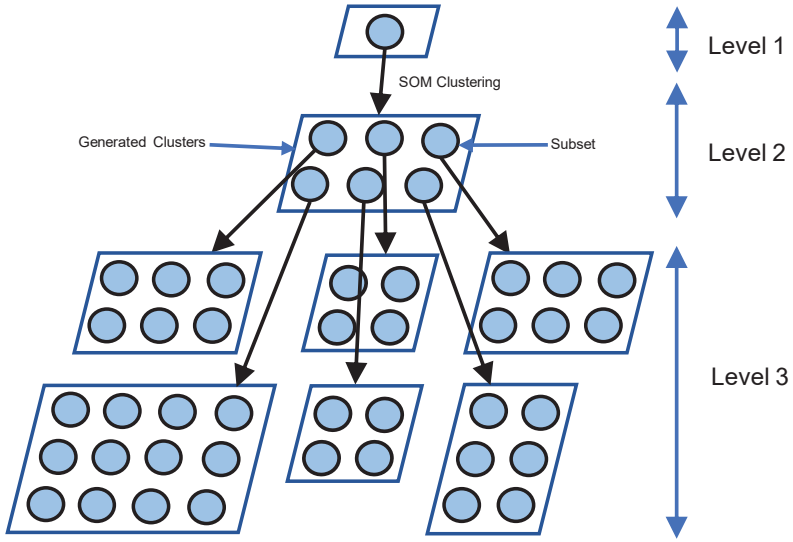


FIGURE 2.3: The Framework of Tree-based SOM

Algorithm 2 TS-SOM

Input: dataset: is the aggregated dataset for weather clustering
θ: represents pre-set thresholds that is used to early stop clustering
attribute: represents attributes of TS-SOM tree, including depth, number of leaves
Output: clusters: are subsets retrieved from clustering
if StoppingCheck(θ, attribute, dataset) **then**
    clusters $\leftarrow$ dataset **return** clusters
else
    nodes $\leftarrow$ SOMClustering(dataset)
end if
attribute $\leftarrow$ attributeUpdate(attribute)
for each node $\in$ nodes **do**
    leaves $\leftarrow$ TS-SOM(node, θ, attribute)
    AddClusters(leaves)
end for
clusters $\leftarrow$ GetClusters() **return** clusters

To proactively reduce prediction errors, a clustering-based model training method is adopted to establish multiple prediction models for various synoptic types. Once the synoptic type is

determined, we can invoke a certain prediction model for further processing. However, there is no standardisation that is globally accepted for weather classification, so that the classification is substantially subjective and cannot precisely reflect the variation of meteorological factors. Thus, weather clustering plays a vital role in establishment of an accurate PV power prediction model. Once temporal pattern aggregation is finished, a dataset containing new feature vectors will be uploaded to the weather clustering module. In this chapter, TS-SOM is employed to realise the weather clustering. It is a hierarchical clustering method to divide the dataset into multiple groups and each node of the tree is mapped to a traditional SOM neural network for further clustering. It enables increasingly fine-grained weather clustering from the root to the leaf. As shown in Figure 2.3, the root of a tree in level 1 contains all the data samples that are involved in performing weather clustering. By implementing SOM clustering on the root, the original dataset is partitioned into different groups and each node on level 2 represents a subset with a unique centroid. In level 3, a new SOM neural network is established for each node in level 2 and the dataset contained in each node of level 2 is further partitioned into subsets of smaller size. Based on these partitioned subsets at level 3, the corresponding child nodes will then grow iteratively. After finishing the training of a SOM neural network in the hierarchical clustering procedure, there will be dead nodes without any data samples. Thus, the actual number of child nodes generated from their direct parents located at the upper layer needs to be adequately adjusted by analysing the feature map of the trained network, where the feature map of SOM reflects the data structure because the corresponding centroids show the distribution of different clusters. To avoid overfitting, pre-pruning is expected to be implemented in TS-SOM. We can set some thresholds for different metrics, including the number of leaves, tree depth, and most importantly, the number of data samples in each leaf. Once any threshold is violated, the newly generated node will be pruned and the tree-building process for the corresponding subtree will be stopped early. The specific description of TS-SOM is shown in Algorithm 2. In this method, the total number of leaves can be recursively retrieved.

Compared with the traditional SOM, TS-SOM enables parallel processing and can be implemented in the edge computing platform, which, meanwhile, helps speed up the training process to a large extent. Besides, this approach is more efficient for clustering hierarchical

data of large size. In general, the weather conditions can be roughly classified into four synoptic types, sunny, cloudy, rainy, and overcast. However, sometimes a partially cloudy day resembles a sunny day for PV power generation. The hierarchical data clustering prevents the case of the incorrect model from being invoked due to the wrong labelling of meteorological data, which effectively enhances the prediction accuracy.

In addition, the weather clustering module is also effective in handling data loss, which is often encountered in PV output prediction. More precisely, the latest meteorological data is not always retrieved on time since inherent and unpredictable network delays could suddenly occur during the data transmission. Once this is imposed, the corresponding model needs to use the centroid of the same cluster as input to make the prediction. If data loss only occurs in partial feature dimensions, KNN and iterative imputation techniques will support inferring the missing part based on the other historical instances of the same cluster. By doing so, a relatively reliable short-term prediction can still be performed without the latest weather data.

2.3.4 Gradient Boosting Tree for PV Prediction

Based on the clustering analysis of the historical meteorological data, the prediction models are established respectively for different clusters. In this study, either XGBoost or LightGBM can be utilised to perform short-term PV output prediction. Similar to the gradient boosting decision tree (GBDT), XGBoost and LightGBM are tree-based models and employ an additive training strategy. They ensemble a set of weak learners generated at different time steps of the training, mostly using Classification and Regression Tree (CART), and the intermediate results are summed up as the final prediction output. Unlike other implementations of boosting algorithm presented in [95] [105], which use pre-sorted algorithms, LightGBM employs a Histogram-based algorithm to compute best split points. For the traditional pre-sorted algorithm, the values of variance gain for all possible split points on pre-sorted feature values need to be calculated, which is inefficient and unexpectedly brings about huge overhead on both computation and storage resources. Instead, a Histogram-based algorithm enhances training efficiency and lowers memory usage to a large extent. Firstly, it maps continuous feature values to discrete values and then splits these values into corresponding bins. Finally,

it uses these bins to construct histograms for all the features during the model training. In each bin, the number of data samples and sum of negative gradients are stored to search for the best split point. Besides, to further enhance the construction speed of histograms in the training process, a histogram differential acceleration strategy is employed. After a node partition is finished, only the histogram construction of a new leaf will bring about computation cost, the histogram of another leaf can be obtained by performing difference between the histogram of its parent node and that of its brother. Instead of growing the tree level wise, like XGBoost [25], LightGBM applies a leaf-wise algorithm to grow the tree. In this manner, the leaf with the better gain will be assigned higher priority to perform node partition whereas a level-wise growing strategy always splits all the nodes located in the same level. The leaf-wise algorithm can provide better performance on accuracy. To further reduce computational complexities when the feature dimension is high and data size is large, two novel techniques, namely Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) are firstly proposed and integrated into the model training procedure. Because instances with large gradients will contribute more to boosting information gain, GOSS randomly drops the instances with small gradients and keeps the instance with large gradients. By using this method, the size of training data is significantly reduced. Regarding the EFB algorithm, the feature dimension can be shrunk by bundling the features that rarely take nonzero values simultaneously. The theoretical analysis of these two techniques is elaborated in [50]. As in the quantitative studies conducted in [50], LightGBM greatly improves the convergence of training and the accuracy of prediction by employing these techniques.

In addition, LightGBM achieves optimisation in parallel learning, which enables distributed model training in different working nodes. A voting parallel algorithm is firstly proposed to better realise a trade-off between communication efficiency and accuracy in LightGBM. For the conventional data-parallel algorithm, in order to globally identify the best split points, working nodes need to communicate with each other to retrieve the histograms of all features generated locally. The communication cost is proportional to the number of features and histogram size. To lower the communication cost, the voting parallel algorithm employs two-stage voting strategies, local voting and global voting respectively. Firstly, top-k features with higher information gain will be selected locally at each working node. Then the

candidate features will be ranked based on the times being selected, and top-2k features will be chosen from the rank list. Within these 2k candidates, the histograms of each feature will be globally aggregated and the information gain of each split point for each feature will be calculated. After realising further global comparisons, the best split point can be identified. The communication cost for this method is O(1). In our work, we integrate the voting parallel algorithm into the edge computing paradigm, and thus the communication cost among edge devices in the model training can be reduced to a large extent.

2.3.5 Interpretability Analysis for the Prediction Model

Unlike the linear models and the rule-based models that directly incorporate interpretability into the structure of the model, GBDT is not an intrinsically interpretable machine learning approach. To obtain its interpretability, we use a Post-hoc global explanation method [33] to understand what knowledge is retained in the model. The feature importance, a widely used metric in the global explanation method, is adopted to explicitly demonstrate the statistical contribution of each selected feature for the prediction. As a type of tree-based ensemble model, there are three traditional feature attribution methods to measure feature importance for GBDT, namely, Weight, Gain and Cover [33]. Weight represents the relative number of times a feature that is used to split across all the ensemble's trees. It indicates how frequently a particular feature appears in the decision trees during the model's training process. Gain measures the average training loss reduction as the gain when using a feature for node partition. Features with higher gain values are considered more informative, as they contribute more to reducing the model's prediction error. Cover represents the relative coverage of the feature across all the trees in the ensemble. It is the percentage of samples that pass through the decision nodes where the feature is used for splitting, relative to the total number of samples in the dataset. Features with higher cover values are used to make decisions for a larger portion of the dataset, indicating their importance in capturing widespread patterns in the data.

However, according to [74], these three methods violate the consistency principle of feature attribution, where the feature with the highest importance will not decrease when the model

changes. Meanwhile, more importance should be attributed to the features close to the root rather than those close to the leaves. LIME (Local Interpretable Model-agnostic Explanations), as an interpretable machine learning technique, provides explanations for individual instances but not necessarily capturing the global behaviour of the model. Since it is based on simple linear models, it may struggle to capture the intricacies of such complex models. Considering such limitations, adopting these methods for measuring global feature importance is not sufficiently reliable. Alternatively, the SHapley Additive Explanation (SHAP) method averages all possible orderings of the features, rather than just the ordering specified by their position in the tree [73]. By doing so, SHAP guarantees that the feature with the highest importance remains unchanged in different regression models to solve the same issue. SHAP is then used as a measurement tool to assess the importance of features within a trained boosting tree, which intuitively demonstrates the learned representations and knowledge.

2.4 Result and Discussion

To evaluate the performance of our proposed framework and tree-boosting method for short-term PV output prediction, we gathered one year of PV power output data from the PV panels and obtained the corresponding meteorological data from the weather station installed at the same location. The data was sampled every 30 minutes from Solcast [1], a platform offering various services and tools to help individuals make informed decisions regarding solar energy generation. Subsequently, we divided the collected records into two distinct portions: 80% for the training set and 20% for the test set. We implemented the computational framework presented in Section 2.3 on both powerful desktop computer and edge devices, and the performance of different models for the PV output prediction is comprehensively evaluated. The experiment results are presented accordingly.

2.4.1 Experimental Setup

To realise evaluation from different perspectives, we set up two testbeds, a powerful desktop computer, and a cluster of edge devices. The desktop computer with Intel i7-7700 (6c/12t)

CPU, 32GB DDR memory, and SSD 2TB 7200rpm hard disk, is utilised to implement three groups of different learning algorithms for PV output prediction respectively. The operating system is Ubuntu 16.04, and the installed software packages for implementing the machine learning algorithms are Python 3.7, Scikit-learn 0.21.3 and tensorflow 1.13.2. The accuracy of different models can be evaluated comprehensively on this testbed.

We also built an edge device cluster that is comprised of three Raspberry Pi 3B, each of which is with a quad-core CPU and 1 GB RAM. The cluster is responsible for comparing training costs of different learning methods on resource-constrained devices. The operating system is 64-bit Ubuntu Mate, and Dask is used as a data analysing framework to build up a cluster and support different workers of a cluster to achieve distributed model training.

2.4.2 Evaluation Metrics

To evaluate the prediction performance of these trained models, we selected three broadly used metrics for measuring regression accuracy, namely, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and coefficient of determination (denoted as "R^2"). The definitions are given below, where $\widehat{y}$ and y denotes prediction value and observed value respectively.

$$\text{MAE } = \frac{1}{N}\sum_{t=1}^{N}|\widehat{y}(t) - y(t)| \tag{2.1}$$

The MAE is the average error value in a set of predictions. This is the average difference between prediction and observed values and all differences are equally weighted.

$$\text{RMSE} = \sqrt{\frac{1}{N} \times \sum_{t=1}^{N}(\widehat{y}(t) - y(t))^2} \tag{2.2}$$

The RMSE examines the root of average squared difference between prediction and observed values. It reflects the quality of a model with non-negative numbers. Values closer to zero are better.

$$R^2 = 1 - \frac{\sum_{t=1}^{N} \left(y(t) - \widehat{y}(t)\right)^2}{\sum_{t=1}^{N} \left(y(t) - \overline{y}\right)^2} \tag{2.3}$$

For the coefficient of determination, the range of "R^2" is between 0 and 1. The closer the value of "R^2" is to 1, the more accurately the model will perform and better generalisation performance is provided. In contrast, if the value is closer to 0, the predictions generated from the model cannot reflect the targets. In addition, to further evaluate the satisfaction on training cost of different models, training time and execution time are another two metrics that will be measured respectively.

2.4.3 Pattern Aggregation Evaluation

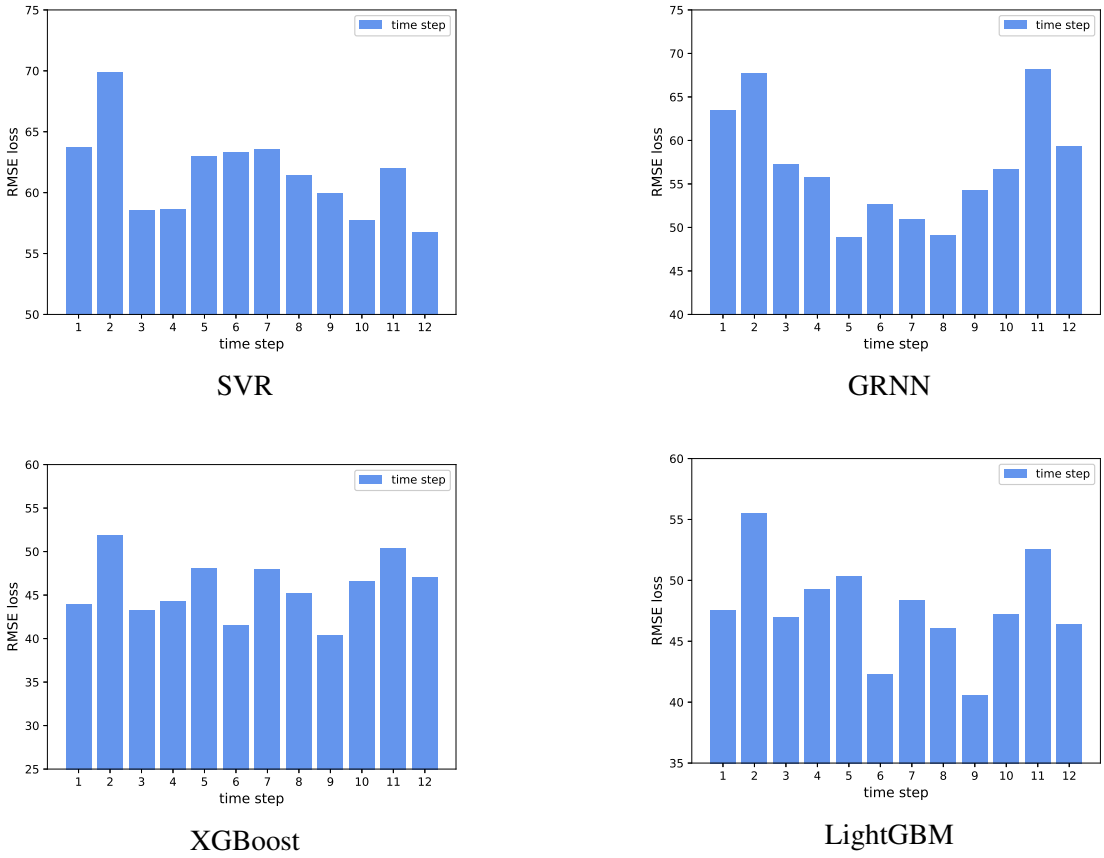
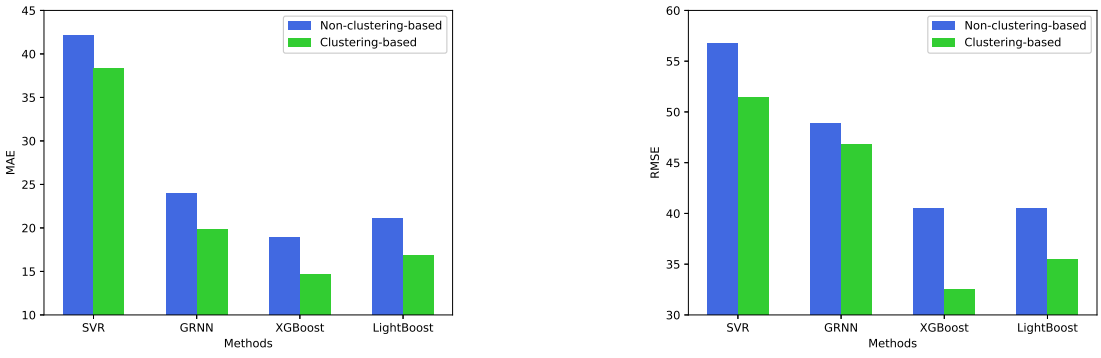


FIGURE 2.4: Accuracy of four prediction models with different sizes of time step

To verify the necessity of temporal pattern aggregation within the model training procedure, we evaluate the performance of different prediction models when the time horizon of temporal pattern aggregation varies. To better illustrate the experiment results, the maximum time horizon for the evaluation is six hours, thus the selected size of time steps does not exceed twelve. The results of performance on accuracy for four models are shown in Figure 2.4. As shown in Figure 2.4, the optimal time-step size resulting in the lowest RMSE value for four models differs. The optimal size of time steps for both XGBoost and LightGBM is nine, which means the accuracy for the predictions is highest when meteorological patterns within four and a half hours engage in and contribute to PV output prediction. In contrast, only the continuous weather feature vectors within five time steps will bring about the best prediction performance for GRNN and the optimal size of time step for SVR is three. After analysing the results, we figure out that the prediction accuracy for the four models will not vary linearly along with the selected time-step size and it is challenging to set up a model to explore optimal size of time steps for different prediction models. Thus it is necessary to implement Bayesian optimisation algorithm to search for the best time-step size for temporal pattern aggregation, especially when the sampling rate of training data is high and the prediction interval is very short.

2.4.4 Prediction Performance Evaluation



MAE Evaluation RMSE Evaluation

FIGURE 2.5: Performance comparison of different methods

TABLE 2.2: Performance of all algorithms

Algorithm	MAE	RMSE	R^2
Group 1: Non-clustering-based			
SVR	42.13	56.74	0.9469
GRNN	24.02	48.94	0.9536
XGBoost	18.94	40.55	0.9712
LightGBM	21.12	40.57	0.9712
Group 2: Clustering-based			
SVR	38.31	51.42	0.9418
GRNN	19.82	46.85	0.9602
XGBoost	14.60	32.58	0.9825
LightGBM	16.79	35.49	0.9792
Group 3: Recurrent neural network			
LSTM	25.12	45.90	0.9652
GRU	25.37	47.85	0.9610

TABLE 2.3: Time Cost of all algorithms

Algorithm	Training(s)	Execution(s)
Group 1: Non-clustering-based		
SVR	0.4343	0.013
GRNN	0.5513	0.207
XGBoost	0.4735	0.00076
LightGBM	0.0540	0.00029
Group 2: Clustering-based		
SVR	0.3130	0.0093
GRNN	0.1960	0.1016
XGBoost	1.0033	0.0067
LightGBM	0.1669	0.0022
Group 3: Recurrent neural network		
LSTM	299.26	20.52
GRU	219.56	21.22

To get a better understanding of the performance and effectiveness of the clustering-based training framework and tree-boosting algorithm, we implemented two boosting trees and another four state-of-the-art methods for PV output prediction, namely GRNN, SVR, LSTM, and GRU. Each method has its own advantages in non-linear fitting and regression problems. We set up three experiment groups for comparison: Group 1: clustering-based machine learning models (SVR, GRNN, XGBoost, and LightGBM), Group 2: non-clustering-based

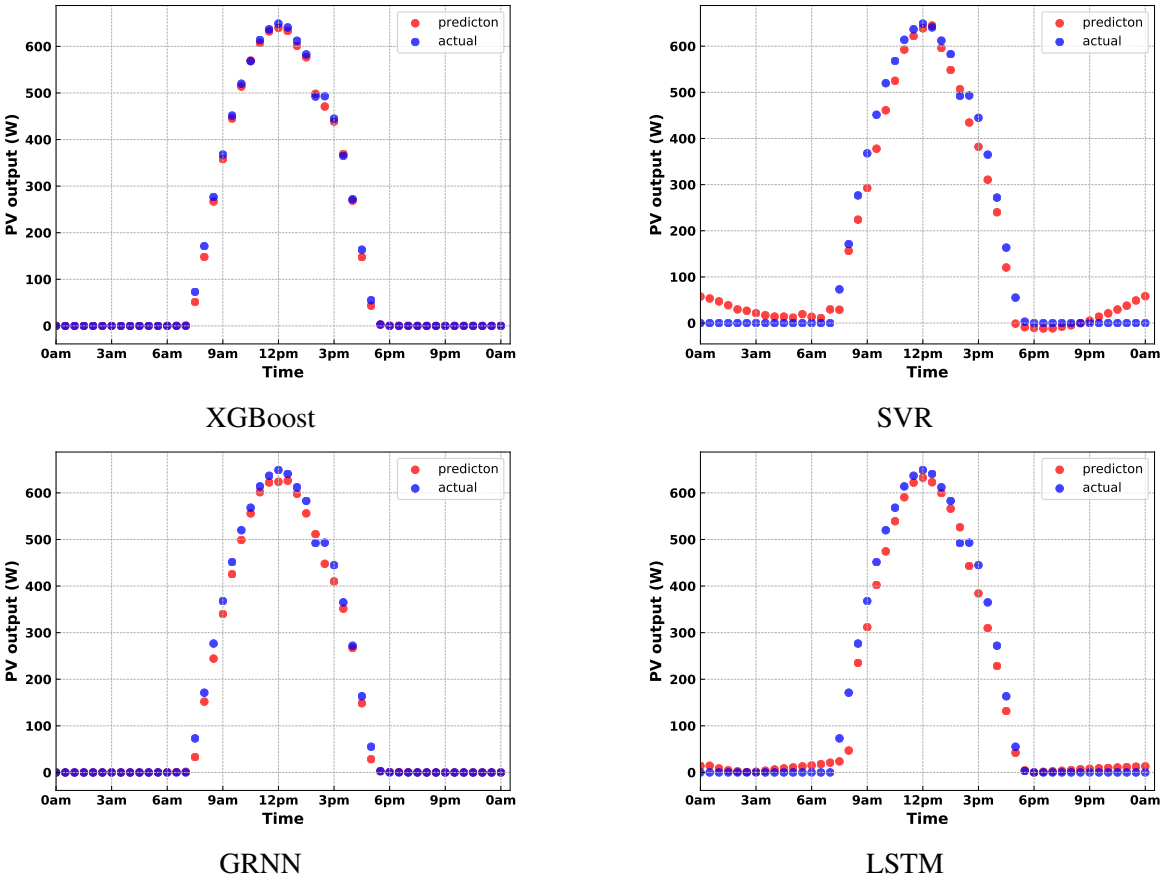


FIGURE 2.6: PV output prediction results of different learning methods for a typical day (randomly selected from the test set)

machine learning models (SVR, GRNN, XGBoost, and LightGBM), Group 3: two types of advanced recurrent neural network (LSTM and GRU). Table 2.2 provides the accuracy results of all methods and Table 2.3 shows the time cost for model training and prediction respectively. Figure 2.5 compares the values of RMSE generated by four prediction models when two different training strategies are employed. The PV output prediction of a typical day using these four models is also given in Figure. 2.6.

As shown in Table 2.2, in both Group 1 and Group 2, two ensemble methods (LightGBM and XGBoost) provide better performance on the test data set with higher accuracy and a lower error rate compared to the others. Besides, with more than 0.97 of R^2, these two models indicate that they can also provide better generalisation performance and are applicable to diverse weather conditions. After investigating metrics of all models, we can see that the

accuracy and applicability of SVR for PV output prediction are the worst. It explicitly indicates that SVR cannot provide good convergence on datasets that exhibit temporal sequence patterns. However, as shown in Figure 2.5, by employing the clustering-based training strategy, the values of RMSE and MAE for SVR model are significantly decreased. Similarly, compared with non-clustering-based models, the accuracy of both clustering-based XGBoost and clustering-based LightGBM is enhanced as well. In contrast, it is interesting to observe that clustering-based GRNN does not show marked reduction on prediction errors. The RMSE of clustering-based GRNN only drops from 48.94 to 46.85.

In addition, to further investigate the performance of ensemble models for PV output prediction, we also implemented LSTM and GRU respectively as a comparative experiment group, because these two deep learning techniques draw increasing attention to copying with temporal sequential problems. For both RNN models, we build up a three-layer network, including input layer, RNN layer and output layer. In the RNN layer, the number of neurons for each LSTM cell and GRU cell is 162 and 189 respectively, which provides the optimal performance on the training dataset. By comparing the accuracy with models in Group 3, two tree-based ensemble models markedly exhibit better performance no matter whether a cluster-based training strategy is adopted or not. Most importantly, the training cost and execution cost of RNN is far higher than XGBoost and LightGBM as shown in Table 2.3. Thus, without a particular cost-efficient distributed training paradigm being employed, these two algorithms cannot support model training in the edge devices. Moreover, we also measured the time cost of making predictions on the test data set for other models. It is noteworthy that even though the performance of XGBoost and LightGBM is similarly good, the time cost of XGBoost is much higher than LightGBM, especially in the model training. Considering limited resources are available on edge devices, LightGBM is more appropriate for use to achieve high device performance.

2.4.5 Performance Evaluation on Edge Devices

To investigate training cost and execution cost of different models on resource-constrained edge devices, we implemented multiple machine learning models on the Raspberry Pi cluster.

TABLE 2.4: Cost of all methods on edge devices

Algorithm	Training time(s)	Execution time(s)
SVR	3.90	0.15
GRNN	2.82	0.407
XGBoost	16.83	0.023
LightGBM	1.39	0.020

As shown in Table 2.4, even the execution time of XGBoost is relatively shorter than both SVR and GRNN, its time cost on training is the highest one of all selected algorithms. Such a result is similar to the experiment result obtained from the tests run on the desktop computer. It is not hard to observe that the training time of LightGBM model is the shortest of all. This is because LightGBM adopts Histogram-based algorithm to compute the best split points for node partition, which enhances training efficiency to a large extent. Moreover, voting parallel algorithm can support to significantly reduce the communication time for all working nodes in the model training. On the other hand, we also evaluated the execution time of the prediction of validation set. Both XGBoost and LightGBM have low execution time cost. GRNN model took the longest time to perform prediction. This is due to GRNN model contains a large number of parameters that is proportional to the amount of training data and complex computation needs to be performed in each prediction. Thus, it leads to high resource consumption on both computation and storage. Based on the analysis of experiment result, we figure out that LightGBM brings the lowest time cost in both training and prediction, and it is appropriate to be implemented on resource constrained edge devices.

2.4.6 Interpretability Evaluation

To evaluate the interpretability of our boosting tree for PV output prediction, we calculated the SHAP values for XGBoost as shown in Figure. 2.7. The features are given in a non-ascending order based on their importance. As shown in Figure. 2.7, the feature importance of "ghi" is the highest, followed by "ghi10" and "ghi90". This indicates that the feature "ghi" makes the most important contribution to the prediction. Apart from the global explanation, Figure. 2.7 also contains the localised explanation for individual weather condition. Each data point in Figure. 2.7 is the Shapley value for an instance of a feature. After analysing the distribution

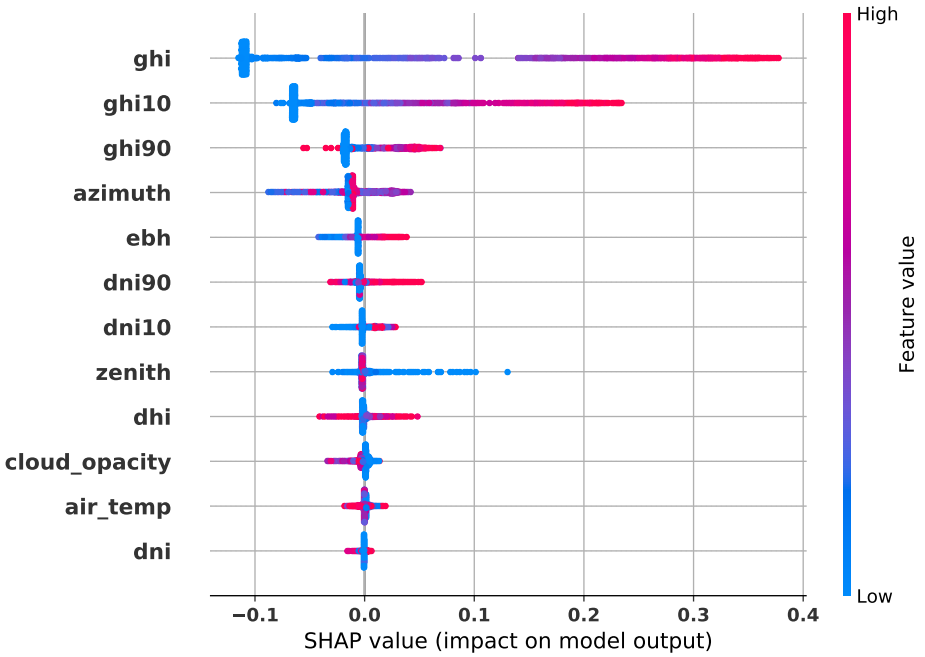


FIGURE 2.7: Inpreterability Analysis Result

density of sample data on each feature, "ghi" clearly makes the most impact on the prediction for most data samples.

2.5 Summary

In this chapter, we proposed an effective computational framework for fine-grained PV power output prediction. In the framework, we integrate temporal pattern aggregation with weather clustering to achieve the reduction of prediction errors. To derive the optimal size of time steps that lead to minimum prediction errors, a Bayesian optimisation algorithm is employed for temporal pattern aggregation. Besides, two tree-structured machine learning algorithms, TS-SOM and boosting tree, are implemented on our designed framework, which are responsible for weather clustering and PV output prediction respectively. To evaluate the performance of our design, we implemented three groups of machine learning algorithms and investigated training costs on different training environments. The experimental results show that our proposed training framework integrated with Lightweight boosting tee can deliver excellent performance on PV output prediction, and is appropriate for implementation on the edge computing paradigm.

CHAPTER 3

Source-free Domain Adaptive Ensemble Trees for Non-intrusive Load Monitoring

---

Sustainable energy management systems have been increasingly studied in recent years. Non-intrusive load monitoring (NILM), as a key component, estimates the power consumption of individual appliances from the main readings only. However, most NILM approaches are computationally expensive, and their generality is negatively affected by the data drift occurred when the models are used across domains. Besides, the threats of privacy violation will rise in the model transfer due to the possible leakage of the personal information of the users from the source domain. To address all these challenges, we designed a cost-efficient learning method using a lightweight implementation of the gradient boosting tree for energy disaggregation. We also proposed a source-free transfer learning algorithm using feature importance analysis, which enhances the generalisation capability of tree-based ensemble models applied in different domains while protecting privacy. We conducted experiments with real-world data sets. The performance of our approach is superior to the state-of-the-art solutions.

3.1 Introduction

The Internet of Things (IoT) enables a wide range of applications in energy management, many of which cannot be best addressed by conventional approaches. Increasing efforts are investing in delivering smart energy management systems for better energy utilisation [83, 48, 131, 109]. Most online energy management strategies heavily rely on accurate predictions of PV output and active incoming loads [72, 60]. At the user end, the information

on energy supply and consumption helps the energy management systems schedule energy-intensive tasks during periods of high solar production. The system can further optimise energy usage by storing excess energy during sunny periods and discharging it during peak load times when solar production is low. Consumers can reap economic benefits by integrating sustainable energy and optimising energy management. Also, such information can help the energy suppliers efficiently schedule power distribution and implement demand response (DR) while consumers' requirements are considered, improving the resilience and reliability of grid power.

Thus, apart from PV output prediction as discussed in Chapter 2, appliance-level energy usage information has been recognised as another key to improving energy efficiency and environmental sustainability. As presented in [124], energy usage behaviours can be derived from the energy consumption of individual devices, but with high costs and privacy concerns. Recent advances in informatics analysis enable deep mining of consumer usage patterns from energy consumption data, reflecting indoor activities in daily life. Previous studies [90, 38, 35] indicate that up to 15% reduction in electricity consumption can be achieved by estimated appliance-level consumption data. Hence, a cost-effective approach to realising the acquisition of appliance-level information needs to be investigated. Non-Intrusive Load Monitoring (NILM) is a cost-effective solution to extract appliance-specific individual electricity consumption from whole-house load readings. Unlike intrusive load monitoring (ILM), NILM only needs a central power for collecting aggregated data, avoiding the high expense of sensor installation and maintenance. NILM is a practical instance of a blind source separation problem where only a single mixture observation is available to recover all individual sources. The uncertainty of energy usage behaviour and unidentifiable operation status of diverse appliances increase the difficulties of appliance-level estimation.

After several waves of technical revolution for NILM, various experimentally feasible schemes have been developed over the years, including hidden Markov model (HMM) and its variants [93] [17], and deep-learning [51] [129] [11]. However, these approaches associate with high computational complexity and training overhead. In recent works [23] [40] [21], GBDT (Gradient Boosting Decision Tree) indicates remarkable performance on computational cost,

accuracy and interpretability for the non-linear regression problems with limited, well-defined feature dimensions. It is therefore worthy of exploring the potential of using GBDT on NILM for better cost-efficiency.

Also, most existing NILM approaches carry out both training and validation in the same domain. The generalisation of those approaches is unknown when crossing different domains without model retraining. Known from our daily life, the signature of any appliance correlates to living habits and operating patterns in different areas. The model trained on the source domain at one place could experience performance degradation in the target domain at another place. More precisely, the captured informative patterns and the knowledge of feature representation kept in the trained model may lose efficacy, as data drift could occur for the changes in feature values and distribution. Most importantly, collecting sufficient labelled data and training a new model for the target domain is very time-consuming and expensive, especially in cold-start scenarios.

Transfer learning is a promising technique to address the above concerns for ML-based applications. It can reuse the knowledge embedded in the trained model by transferring the well-trained model to the target domain with the minimum amount of retraining [91]. The new model will adjust in the target domain with new data to provide a similar or even better performance of energy disaggregation. Transfer learning can significantly reduce the training time and computational resources required for modelling, and accelerate the development and deployment of real-world NILM applications.

In previous studies, numerous transfer learning techniques [34] [139] [5] [69] had been developed based on neural networks for NILM applications. However, the lack of an effective method for domain adaptation precluded the use of GBDT in NILM applications. It is thus essential to develop a transfer learning algorithm for the tree-based models. In addition, [80] indicates that sensitive and private information can be extracted from historical energy readings with advanced data mining techniques. Traditional transfer learning approaches, e.g. instance-based [53], feature-based [5] and relation-based [86], could expose the readings of the source domain and leak the privacy of end-users. To this end, this chapter proposes a source-free model-based transfer learning approach for a tree-based ensemble model to enable

accurate energy disaggregation across different domains in practical use while addressing privacy concerns. The model transfer only requires the data at the target scene without any source data involved. In addition, the model-based approach retains the entire architecture of the pre-trained model from the source domain, and only the partial weights and parameters may be further fine-tuned on the limited target data for new tasks. As a result, the approach significantly reduces computation and storage while leading to improved performance, faster convergence, and enhanced generalisation.

The main contributions of this chapter are:

(1) We unprecedentedly integrate an ensemble learning approach with the sequence-to-point optimisation for NILM by only analysing low frequency power reading.
(2) We detect the changes in statistical properties of two different domains by performing interpretability analysis on the trained NILM models. Based on the interpretability analysis result, we discuss the influence of such changes on the estimations and develop an algorithm to reduce the risk of data drift during the model transfer.
(3) We propose a source-free transfer learning approach with joint consideration of effectiveness and privacy for any tree-based ensemble model.
(4) We conduct a comprehensive evaluation of the tree-based ensemble models for NILM application and validate the feasibility of the proposed transfer learning framework.

3.2 Background

Many studies have been done on energy disaggregation since NILM proposed in the 1980s, while HMMs were the most favoured technique for the earlier works. A recent work [93] developed an HMM-based energy disaggregation approach without a complete knowledge of its load devices. An unsupervised approach, with non-parametric FHMM, achieved energy disaggregation with low-sampling energy reading [47]. However, [118] argued that the efficacy of HMMs reduces as the size of the problem increases due to the presence of local optima. [76] introduced another energy disaggregation algorithm that preserves dependencies between energy loads with variants of HMM and viterbi algorithm. This algorithm is efficient

enough to achieve real-time energy disaggregation with high accuracy results. However, the complexity of this algorithm grows exponentially with the increase of the number of appliances, which limits its practicality and suitability. In addition, the study [55] pointed out the flexibility of this algorithm is limited, due to sub-meters need to be installed weeks ahead when it deploys to new scenarios. [78] showed the performance of HMM can degrade due to the load features of target appliances. The later work [37] [31] [88] [45] [79] [129] [11] used deep-learning techniques for energy disaggregation. Some of them [45] [79] reformed the HMM by performing deep-learning approaches as the feature extractors, while other works obtained energy disaggregation results directly with deep-learning approaches [129] [11]. [129] proposed a Convolutional Neural Networks (CNNs) based sequence-to-point approach to estimate the power consumption of the midpoint of an input sequence, which outperformed the existing sequence-to-sequence approaches. However, most deep learning methods significantly increase the training overhead, so they are hard to complete model training on devices with limited computation and memory resources in a reasonable time.

To further reduce the expense of obtaining labelled samples from various domains, [101] designed a data augmentation approach to improve the transferability and accuracy of DNN (deep neural network)-based NILM models. [34] [139] [5] [69] explored using a transfer learning scheme to apply the models trained on extensive data sets to different target domains. [34] extended [129] by adopting a transfer learning scheme to build an energy disaggregation model with a low sampling rate and limited training samples. [139] [5], and [69] adopted the state-of-the-art neural networks, generative adversarial networks and probabilistic neural networks to realise energy disaggregation. Domain adaptation is performed to narrow the domain shift between the synthetic and real-world domains. For NN-based models, the performance of traditional transfer learning techniques, such as parameter sharing and fine tuning, highly depends on the similarity of statistical patterns across different domains [5]. So TrGAN in [5] and GADM in [69] employed feature-based approaches and improved the performance by minimising statistical distance. However, the historical data from the source domain is necessarily involved in the transfer learning. Such involvement could lead to potential privacy leakage since [80] has demonstrated that the sensitive information is able to extract from disaggregated energy loads. It is thus essential to study how to improve the

performance of domain adaptation without leaking sensitive information from the source domain.

3.3 Methodology

This section will specify all the techniques and our developed algorithm to enable cost-efficient model training for the NILM problem and solve the performance degradation due to data drift on new scenes. We will first give an introduction to LightGBM and the sequence-to-point learning paradigm. The integration of these two techniques guarantees high accuracy of energy disaggregation. Then, the data drift between different domains is theoretically and empirically investigated using a feature importance measurement. We finally present a source-free model-based domain adaptation method to realise model transfer based on the feature importance results.

3.3.1 LightGBM for NILM

Tree-based ensemble models show consistently high performance and cost-efficiency on nonlinear regression problems, particularly when the interpretability and hardware cost of parameter tuning are counted. As a variant of GBDT, LightGBM is a tree-based ensemble model and optimises training speed and memory consumption by using the histogram-based algorithm. Due to its superiority of training efficiency against other methods, we use a LightGBM as the learner for solving NILM. Similar to the original GBDT, LightGBM uses an additive training strategy [50]. It integrates a set of weak learners iteratively generated at different time steps of the training, mostly using Classification and Regression Tree (CART), and the intermediate results are summed up as the final prediction output. In regression, the objective function of LightGBM is defined as:

$$obj_{(t)} = \frac{1}{n}\sum_{i=1}^{n} l(y_i, \hat{y_i}^{(t-1)} + f_t(x_i)) + \Omega(f_t). \tag{3.1}$$

where x_i is a feature vector of the i^{th} sample data, and y_i is the corresponding label, which is the actual power consumption of the target appliance in this study. $\hat{y_i}^{(t-1)}$ represents the

prediction result of x_i at time step $t-1$ in the additive training process of LightGBM, and the function f_t is a new CART generated at the t^{th} iteration, which is responsible for mapping a certain training sample x_i to the corresponding leaves. The function l is used to calculate the squared error of each data sample. $\Omega(f_t)$ represents the regularisation term used to combat overfitting of the new tree. The regularisation term imposes complexity constraints on the tree by limiting the number of leaves and the scores on leaves. It can be mathematically defined as:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{3.2}$$

$$w_j = \frac{(\mid \sum_{x_i \in j} g_i \mid -\gamma)^2}{\sum_{x_i \in j} h_i + \lambda} \tag{3.3}$$

where T is the number of leaves, w_j represents the score assigned to the leaf j, and γ and λ are hyperparameters of the penalty terms. w_j can be calculated by Eq.(3.3), where g_i and h_i represent the first-order and second-order derivatives of a sample x_i, allocated to the leaf j. Similar to GBDT, the feature that has the highest variance gain will be selected for each node partition of a tree generated at the t^{th} iteration. The formal mathematical definition used to calculate the variance gain for each partition in LightGBM is given by:

$$\begin{aligned} Gain(O, X^k, S) = &\sum_{x_i \in O} l(y_i, \hat{y_i}^{(t-1)} + w^p) - (\sum_{x_i^k \leq S} l(y_i, \\ &\hat{y_i}^{(t-1)} + w^l) + \sum_{x_i^k > S} l(y_i, \hat{y_i}^{(t-1)} + w^r)) \end{aligned} \tag{3.4}$$

where O is a set of samples to be partitioned at the tree node o, X^k denotes the k^{th} feature, S is the threshold for the feature X^k splitting the training data, x_i^k denotes the value of feature X^k for the i^{th} data sample, w^p is the score value assigned to the parent node o, and w^l and w^r represent the scores assigned to the left leaf and right leaf respectively. Based on Eq.(3.4), the best split points $\{X_{best}^k, S_{best}\}$ are expected to be found in the tree growth procedure.

3.3.2 Structuring Temporal Sequences

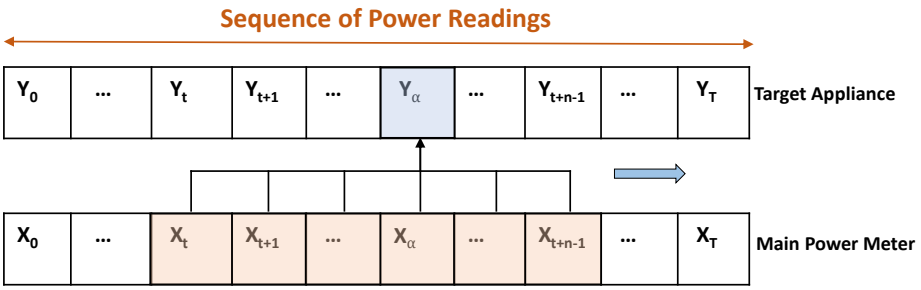


FIGURE 3.1: Sequence to Point

NILM aims to recognise the working status of any appliance from aggregated power measurements and determine the power consumption at given time points. The inputs used for NILM model training are continuous time-series aggregated load measurements from the main power meter. When the readings are in high frequency, the computational complexity and memory cost will significantly increase in both processes of training and inference. It is also hard to select a regressor that can comprehensively capture operating patterns of appliances and keep the learned knowledge into a model when the input is dense. To address these issues, we split a long time-series of readings into small sequences that contain appliance usage patterns in limited time steps. In this work, we manage the number of features by shortening the length of the input while keeping the critical information and knowledge for learning energy consumption patterns of each appliance. A sliding window is used to generate a set of short sequences from a long measurement as samples for the NILM model construction.

Considering LightGBM only supports single value regression output, it is hard to apply traditional sequence-to-sequence methods in our training framework. Inspired by the sequence-to-point learning approach in [129], we map a short sequence of data to the corresponding label at the midpoint of the given sequence. For example, as shown in Figure 3.1, the input is a fixed-size window of the main power readings $X_{t:t+n-1}$, where t is the start time point of the sliding window and n is the size of the window. The output is the power consumption of a target appliance at the midpoint of the corresponding window, denoted as Y_α, where $\alpha = t + \left\lfloor n/2 \right\rfloor$. The window slides from the beginning of the sequence to the end, and the step size is 1. In this process, $X_{t:t+n-1}$ can be deemed as a feature vector for the label Y_α and

we proposed to train a regressor F which maps $X_{t:t+n-1}$ to Y_α. The regressor is defined as:

$$Y_\alpha = F(X_{t:t+n-1}) + \epsilon \tag{3.5}$$

where ϵ is the expected error of the estimation. To handle the start points and end points of the output sequence Y=$\{Y_0...Y_T\}$, we use two strategies to construct the sequence. If the scale of data set is large enough, we will generate full feature vectors for all elements of Y by using the padding technique introduced in [129]. In this strategy, we first pad input sequence X with $\left\lceil n/2 \right\rceil$ zeros at the beginning and end, and then corresponding feature vectors with zero values can be generated for all elements at the edge of Y. These data samples will become noise in the model training to help prevent overfitting. However, if the collected data set is small, which is used as the target domain in the model transfer, it is rational to abandon the start points and end points of Y because these samples with incomplete information will significantly reduce the accuracy of the new established model. Thus, in this case, only the element Y_α of the output sequence, where $\alpha \in (\left\lfloor n/2 \right\rfloor, T - \left\lceil n/2 \right\rceil + 1)$, is considered.

3.3.3 Feature Importance Analysis

There is always an intuition behind NILM, which is the appliances of the same type share similar signatures that show the typical operating regime, such as active power and state transition edges. In a static and ideal environment, the trained model is adequate for different scenes and provides accurate estimation if the signatures of the appliances remain unchanged. However, the statistical properties of the variables for energy disaggregation, such as value scale and shape of distributions, could react to the environmental variations. With rapid hardware development, the same appliance from multiple generations could behave differently. Similar changes could also happen as the lifetime of an appliance decreases. The instances distribution of an appliance relies on living habit and usage patterns of the users in different regions. Without learned knowledge transfer functions, the trained model would perform worse in another region for most of the times.

In this work, we investigate how the informative patterns change from the source domain to the target domain, and to what extent the changes affect the estimations. The change of

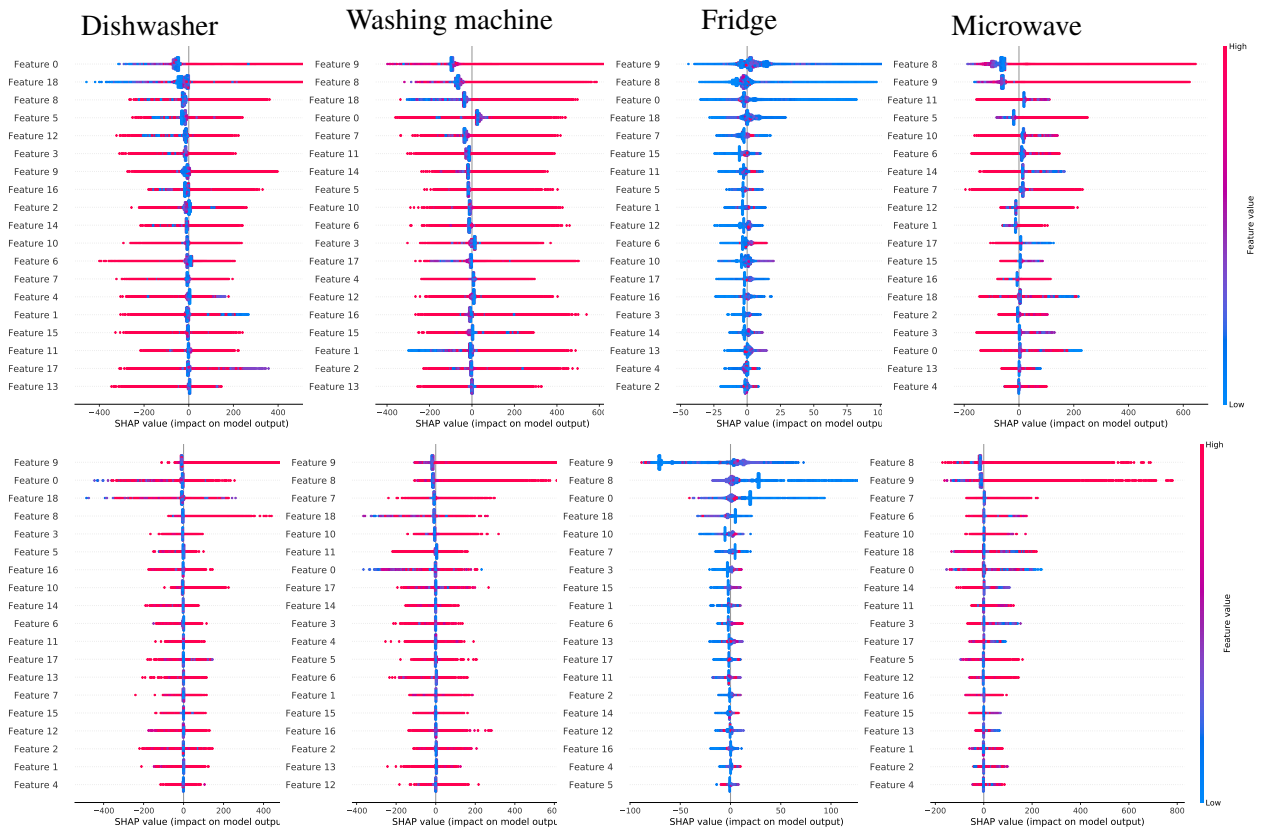


FIGURE 3.2: Comparison of SHAP feature importance of different appliance models on UK-DALE (top) and REDD (bottom)

statistical properties involved in the new data is defined as data drift. The best approach of implicitly detecting data drift is to monitoring the correlation between predictions and selected features. Traditional approaches for correlation analysis, including Pearson, Kendall, Spearman and Information Value, either apply to linear correlation or only accept input data that must be category features. We instead use feature importance to reveal the correlation between a label and a feature. Shapley Additive Explanation (SHAP) method is used to measure the feature importance in a trained model where the importance of each feature is given by the Shapley value. SHAP enables both global interpretations and individual interpretations. Meanwhile, it guarantees consistency property for feature attribution [73]. By doing so, SHAP implicitly shows the variations of learned representations and knowledge between two different domains.

For ease of interpretation, we use two widely used open data sets, UK-DALE [52] and REDD [54], to demonstrate feature importance analysis. Figure 3.2 shows four groups of SHAP summary plots of four appliances, namely dishwasher, washing machine, fridge and microwave. In a summary plot, the features are ranked in descending order by their average absolute Shapley value. Each feature represents the power reading at a time point within a fixed-size sliding window. Each point on a summary plot represents a sample instance, and its position in x-axis depicts the corresponding Shapley value for that feature. The distribution of the Shapley values per feature for all instances can be obtained in a plot. As shown in Figure 3.2, the importance of the features of the same appliance could vary considerably from one domain to another. For the dishwasher model, the feature 12 is ranked as a top-5 feature on UK-DALE, whereas losing its statistical significance on REDD. The feature 9 is the most important feature on REDD while it is not ranked in top 5 in UK-DALE. For the microwave model, feature 11 and feature 5, both ranked as top 5 features on UK-DALE, but show low magnitudes of feature attributions on REDD. Similar to this one, for the fridge model, the importance of feature 10 and feature 11 varies in between two domains. It is noteworthy that the scale of SHAP value for some features and the relationship between feature value and the corresponding impact on the model output could still vary even their ranking remains unchanged. For example, feature 0 and feature 18 in the dishwasher model show a positive correlation with the model output on UK-DALE. However, massive sample instances of these two features make negative contribution to predictions when the model migrates to REDD.

Despite the large differences mentioned above, according to Figure 3.2, it is also interesting to notice that the features, making more impacts on estimations in both domains, are either the central points or the edge points of the sliding window, such as feature 9, feature 8, feature 0 and feature 18. For the models of fridge and washing machine, the important features remain largely intact in both domains. Moreover, feature 8 and feature 9 dominate the feature effects on different data sets. Thus, the feature importance analysis reveals the appliances of the same type from different domains share more or less similar patterns, which helps us to realise model transfer in real-world applications.

Algorithm 3 Feature Selection for The Target Domain

Input: M_s: is the trained LightGBM model for the source domain
$SHAP$: is the function to derive a list of features with
information of SHAP values for a given data set and model
D_t: is the data set from the target domain
λ: is the threshold to select features
Output: L'_{shap}: is the final list of all selected features

$L'_{shap} \leftarrow []$
$L_{shap} \leftarrow$ SHAP(M_s, D_t)
$L_{shap} \leftarrow$ Sort(L_{shap})
$S_{total} \leftarrow$ SumShap(L_{shap})
ratio $\leftarrow 0$
for feature $\in L_{shap}$ **do**
    ratio$\leftarrow$ ratio+ feature.shap/ S_{total}
    if ratio $\leq \lambda$ **then**
        L'_{shap}.append(feature)
    else
        break
    end if
end for
return L'_{shap}

3.3.4 Model-based domain adaptation

Although LightGBM is a cost-effective machine learning approach for NILM, it is arguable that the trained model can transfer to different regions or appliances for sustainable use. Direct sharing is the most common way for implementing model transfer between two domains, but it could degrade the accuracy of estimations when the source and target do not share the same input patterns. However, if we train a new model, a small data set makes the model more susceptible to overfitting leading to poor generalisation. The poor generalisation will cause accuracy loss on new instances that do not appear in the training data set. It will also cause the extra computational cost for model training if the size of the data set of the target domain is larger. The data of energy consumption of appliances is always sensitive because human behaviours and usage patterns can be extracted from it. To protect the privacy of end-users from the source domain and alleviate data drift, we develop a model-based transfer learning algorithm, other than the instance-based or relation-based approaches.

Algorithm 4 Model-based domain adaptation

Input: $TrainGBM$: is the lightGBM training function
M_s: is the trained LightGBM model for the source domain
T_s: is the number of trees of the trained model for the source domain
D_t: is the data set from the target domain
T_{max}: is the maximum number of trees in the target model
L'_{shap}: is a list of selected features that keep unchanged
P_t: is the pre-set parameters for continually training
σ is the parameter that activates early stopping
γ: is the threshold for revising the target model
Output: M_t: is transferred model for the target domain
function TREEREVISE(G′, H′, node, D_{sub}, L'_{shap}, γ)
  if node.splitFeature is null or γ is not satisfied **then**
    node.weight← CalculateWeight(G′, H′, D_{sub})
    node.SetAsLeave()
  else
    node.splitFeature←FindBestSplit(G′, H′, D_{sub}, L'_{shap})
    D^l_{sub}, D^r_{sub}, ←DataSplit(node.splitFeature, D_{sub})
    G′$_l$, H′$_l$, G′$_r$, H′$_r$, ←DerivativeSplit(node.splitFeature,
    G′, H′)
    TreeRevise(G′$_l$, H′$_l$, node.leftChild, D^l_{sub}, L'_{shap}, γ)
    TreeRevise(G′$_r$, H′$_r$, node.rightChild, D^r_{sub}, L'_{shap}, γ)
  end if
  return node
end function
$M_t \leftarrow$ []
G′, H′= Initialise(M_t, D_t)
for tree $\in M_s$ and EarlyStop(σ, M_t) is false **do**
  root′ ← TreeRevise(G′, H′, tree.root, D_t, L'_{shap}, γ)
  tree′ = ConstuctTree(root′)
  M_t.append(tree′)
  G′, H′= UpdateDerivative(M_t, D_t)
end for
T_t=M_t.GetTreeNum()
if $T_t < T_{max}$ **then**
  $M_t \leftarrow$ TrainGBM(D_t, P_t, T_{max} - T_t, M_t)
end if
return M_t

In our approach, we iteratively modify split points of each tree to achieve domain adaptation. Compared with [36], our proposed algorithm enables adaptive feature selection in a new domain rather than just retaining the entire backbone of the source model. More precisely, if the best split feature for nodes from the original LightGBM model loses its efficacy on

the new data set, all features need to be reconsidered. Thus, there are two key steps in our design. Before modifying the original model, we evaluate the importance of features on the new data set and select the features that make more contributions to the estimations on the target domain. As shown in Algorithm 3, the features in the target domain will initially be ranked by SHAP values, and then the top features in the ranking list will be iteratively pushed into a new feature list L'_{shap} based on a pre-set threshold λ. Once L'_{shap} is determined, the original model needs to be modified according to the data set D_t from the target domain. Algorithm 4 illustrates the general workflow of model-based domain adaptation. All element trees within the trained model M_s are modified iteratively and then grouped together as a new model. Function TreeRevise() is called to recursively access all nodes of the tree and adaptively revise their parameters. In this procedure, four scenarios need to be addressed by different strategies.

(1) **Parameters tuning for intermediate nodes**: As mentioned, feature importance could vary with the change of feature distribution and the value scale between two different domains. The original split features that lose efficacy on the target domain cannot provide sufficient informative patterns to support a remarkable increase on the split gain at a given node. To maintain the performance of the model for energy disaggregation, we need to reserve the split features involved in L'_{shap} and pay more attention to the split features that lose power for the estimation. Besides, to avoid unbalanced sample division due to the change of feature distribution and value scale, it is also necessary to find the best split points for all intermediate nodes. Based on Eq.(3.4), the gain of each possible split point can be calculated. In this process, both first-order and second-order derivatives of all samples, G′ and H′, are constantly updated based on the output of the current target model M_t. To improve the training efficiency, the histogram differential acceleration strategy used in LightGBM is also employed in the searching process. We do not need to evaluate the split gain for all possible feature values. Like the tree construction, the computational complexity for split finding is proportional to the size of the data set and feature dimensions.

(2) **Score calculation for leaves**: For a LightGBM model, the predictions of data samples are determined by the leaf scores of each tree member, and each leaf score

is calculated by the training samples allocated to that leaf. Considering that the split point of each intermediate node and distribution of input data has changed in the revise of each tree, the samples fall into a given leaf node will become different. When a leaf node is accessed, a new leaf score is expected to be calculated by Eq.(3.3) and then assigned to the corresponding leaf.

(3) **Pre-pruning for trees**: LightGBM is a leaf-wise tree-structured model, and hence the depth of some branches is greater than others. If the whole backbone of the source model is strictly reserved, the training data from the target domain will not be divided in the way as expected in the source domain because of impacts from data drift. Imbalance division of data will occur so that the number of samples allocated into some nodes may be extremely small. In this case, two potential issues should be taken into account. First, the continuous division will lead to overfitting, which hinders the model providing high performance as expected in the future scenes. On the other hand, for nodes deep down the tree, it is hard to find a split point that provides positive information gain. To prevent overfitting and make better use of statistical information of collected target samples, we impose complexity constraints γ on the currently revised tree by limiting information gain, the number of leaves and the number of training samples in each node. These well-defined constraints are hyper-parameters that need to be provided at the beginning of the model transfer process. In practice, it is required to set these parameters adaptively on the basis of the scale of the training set. Once any of these pre-set constraints are violated, the currently visited node will be set as a leaf, and the subtree from this node will be removed.

(4) **Abandon invalid trees**: In the iteration of domain adaptation, the parameters of intermediate nodes, leaf scores and even tree structure of some visited trees have been revised. This will possibly take effects on the following trees that have not been visited, and these trees maybe lose efficacy to provide positive information gain. Instead, continuous revision of these trees will possibly pose threats to decreasing the accuracy of the target model. Therefore, we will keep eyes on the trend of the improvement of the information gain, as shown in line 25 from Algorithm 4. If

the gain of M_t keep unchanged or drop down in σ iterations, we will terminate the domain adaptation, and abandon the trees of the original model.

After modifying all element trees, continual training is performed if the maximum number of trees T_{max} is larger than the number of trees T_s in the source model. LightGBM training module in API is called, and the wrapper function TrainGBM() will train T_{max} - T_t trees based on parameters P_t and modify target model M_t accordingly.

3.4 Experiment and Discussion

This section implemented our design and two state-of-the-art models, CNN-based and GAN-based, to evaluate NILM disaggregation performance[1]. To promote a fair comparison and reproducibility of results, we used two popular data sets, UK-DALE and REDD, for our experimental studies. We also validated the effectiveness of the model transfer by comparing the accuracy of transferred LightGBM models with and without our proposed algorithm. The model transfer was also implemented on the CNN-based and GAN-based models as performance indexes.

3.4.1 Data Sets

In our experiments, we chose the readings of four common appliances, washing machine, fridge, dishwasher, and microwave in REDD and UK-DALE, for NILM analysis. Our experimental data sets were built by the power readings from building 1 and 2 from UK-DALE and houses 1, 2, 3 from REDD. They were used to evaluate the performance of LightGBM for the NILM application and perform comparisons with other learning methods. REDD and UK-DALE are collected from two different countries, showing different statistical patterns in practical use. They are thus suitable for validating the feasibility of the proposed transfer learning approach in real-life NILM applications. We created the models on UK-DALE and transferred them to the target domain represented by REDD since the number of

[1] https://anonymous.4open.science/r/Transferable-Tree-based-Ensemble-Model-for-NILM README.md

samples of UK-DALE is 350K, far more than REDD with only 10K samples. The sampling rate of readings from two data sets is different, so it is critical to pre-process the data and make their sampling rate the same before further processing. For each data set, 80% of samples were used for model training and the remainder for testing.

3.4.2 Experimental Setup

To conduct our experiments, we implemented multiple training models on a desktop computer with Intel i7 8700 Processor, 32GB DDR4 RAM, and NVIDIA GTX 1080 Graphics Card with 8GB GDDR5X VRAM. The operating system is Ubuntu 16.04, and the software packages used for the experiments include Python 3.7, Scikit-learn 0.21.3, TensorFlow 2.30, Keras 2.4.0 and LightGBM 3.0.0. In this work, we used two state-of-the-art models as comparison baselines. One is CNN integrated with sequence-to-point learning paradigm, and the other is GAN trained with sequence-to-sequence learning paradigm. Both showed high accuracy in previous NILM studies against other non-ensemble machine learning approaches while providing network interpretability by visualising the operation patterns captured from feature maps. We resembled the architecture of the neural network and the hyper-parameters for model training in [34] and [5]. The setting of hyper-parameters for different models is given in TABLE 3.1. The window size used in our experiments was set to 19 by default unless otherwise indicated. [103] specifies the impact of the window size on disaggregation performance and provides an assessment measure to guide the window size selection.

3.4.3 Performance Metrics

In our experimental studies, five widely used metrics were used to evaluate the performance of energy disaggregation designs from different perspectives. All the tests were evaluated on a time series, ranging from 1 to T. At a given time point t $\in [1, T]$, $\widehat{y}_t^i$ and y_t^i represent the power consumption estimation and ground truth power of the appliance i, respectively. The details of the selected metrics are provided below.

TABLE 3.1: The parameters for training NILM models

Hyper-parameters for training LightGBM

Maximum boosting round	100	Learning rate	0.232
Maximum depth	10	L1 regularisation	0.0214
Maximum bins	500	L2 regularisation	0.0001

Hyper-parameters for training CNN

Maximum epochs	100	Learning rate	0.001
Batch size	1024	Beta1	0.9
Early-stopping epochs	10	Beta2	0.999
Optimizer type	Adam	Epsilon	10^{-8}

Hyper-parameters for training GAN

Maximum epochs	100	LR for generator	0.00002
Batch size	1024	LR for discriminator	0.00001
Early-stopping epochs	10	Beta1	0.5
Optimizer type	Adam	Beta2	0.999

The accuracy of the power consumption estimate for an individual appliance is often measured by MAE and Signal Aggregate Error (SAE) [93] [34]. By averaging absolute differences between $\widehat{y}_t^i$ and y_t^i, MAE illustrates the deviation between the estimates and observed values, while SAE measures the relative error between the total energy estimate and actual energy consumption of an individual appliance. SAE is defined as

$$\text{SAE} = \frac{|\sum_{t=1}^{T} \widehat{y}_t^i - \sum_{t=1}^{T} y_t^i|}{\sum_{t=1}^{T} y_t^i} \tag{3.6}$$

For MAE and SAE, their value scales differ for different appliances. It is hard to tell whether the differences reflect a more, or less, model performance. To tackle this issue, we used the normalised disaggregation error (NDE) [56] to compare the accuracy of the models used for different appliances. It reports the normalised error between the estimates and the ground truth of an appliance and is defined as

$$NDE = \sqrt{\frac{\sum_{t=1}^{T} \|\hat{y}_t^i - y_t^i\|^2}{\sum_{t=1}^{T} \|y_t^i\|^2}} \tag{3.7}$$

TABLE 3.2: Different Models Tested on UK-DALE

Appliance	CNN (seq2point)			LightGBM			GAN		
	MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE
Washing machine	17.708	0.011	0.514	17.110	0.0034	0.519	18.592	0.635	0.366
Fridge	20.873	0.022	0.533	18.861	0.0003	0.509	34.107	0.334	0.609
Dishwasher	20.786	0.176	0.492	17.857	0.0042	0.503	14.344	0.105	0.105
Microwave	8.742	0.127	0.658	6.886	0.0025	0.671	5.793	0.158	0.397
E_{Acc}		0.700			0.728			0.670	

For all three metrics, the lower the value, the more minor deviation between estimates and ground truth generated by the model. As mentioned, the performance of a NILM model varies among appliances. It is also essential to introduce a metric to evaluate the overall performance of an algorithm. In this work, the estimation accuracy E_{Acc} [54] is used and given by

$$E_{Acc} = \left[1 - \frac{\sum_{t=1}^{T}\sum_{i=1}^{N}|\hat{y}_t^i - y_t^i|}{2\sum_{t=1}^{T}\sum_{i=1}^{N}|y_t^i|}\right] \tag{3.8}$$

Unlike the metrics aforementioned above, a NILM method is verified to provide good overall performance only if E_{Acc} is low. We also used two other metrics [100], estimated energy fraction index (EEFI) and actual energy fraction index (AEFI) in our experimental studies. They are used to demonstrate the overall performance of various algorithms and illustrate the deviation between the estimated energy fraction and actual energy fraction of an appliance. They are defined as

$$EEFI = \frac{\sum_{t=1}^{T}\hat{y}_t^i}{\sum_{t=1}^{T}\sum_{i=1}^{N}\hat{y}_t^i} \tag{3.9}$$

$$AEFI = \frac{\sum_{t=1}^{T}y_t^i}{\sum_{t=1}^{T}\sum_{i=1}^{N}y_t^i} \tag{3.10}$$

3.4.4 Performance of LightGBM

To examine how LightGBM is performed on NILM, we compared it with the CNN-based and GAN-based solutions from different perspectives. All models were trained and tested on the same data set to allow fair comparison. TABLE 3.2 and 3.3 shows how well these models

TABLE 3.3: Different Models Tested on REDD

Appliance	CNN (seq2point)			LightGBM			GAN		
	MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE
Washing machine	4.590	0.028	0.194	4.755	0.0071	0.228	6.976	0.140	0.300
Fridge	33.006	0.006	0.560	26.300	0.0014	0.480	55.234	0.931	0.940
Dishwasher	16.909	0.084	0.635	13.417	0.0060	0.588	12.125	0.472	0.730
Microwave	17.150	0.288	0.611	12.431	0.0057	0.592	13.201	0.716	0.951
E_{Acc}	0.689			0.757			0.613		

TABLE 3.4: Performance of Transferred Models on REDD

Appliance	CNN (Domain Adapted)			LightGBM (Directly Transferred)			LightGBM (Domain Adapted)			GAN (Domain Adapted)		
	MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE
Washing machine	5.008	0.006	0.205	28.034	0.498	0.848	3.265	0.0043	0.149	21.508	0.715	0.817
Fridge	30.525	0.085	0.551	53.587	0.303	0.749	29.753	0.0013	0.528	51.249	0.833	0.895
Dishwasher	16.498	0.001	0.620	23.146	0.439	0.850	11.741	0.0081	0.508	15.011	0.939	0.960
Microwave	12.848	0.225	0.611	21.061	0.003	0.905	10.521	0.0062	0.524	16.658	0.607	0.947
E_{Acc}	0.716			0.449			0.762			0.528		

TABLE 3.5: Comparison of average overhead of three models for UK-DALE, including model size, training cost and inference cost

Method	**Model Size(KB)**	**Training Time(s)**	**Inference Time(s)**
CNN	11935	5621.41	82.50
LightGBM	830	162.42	8.09
GAN	182896	11722.01	212.40

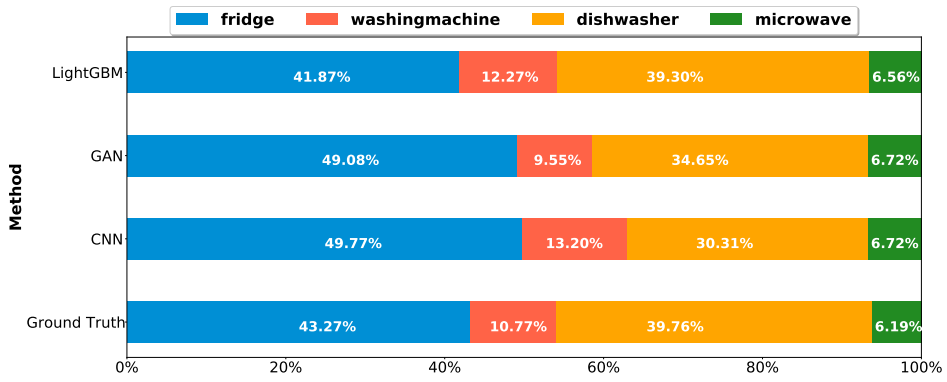


FIGURE 3.3: EEFI and AEFI on UK-DALE

performed on UK-DALE and REDD respectively. By comparing the values of MAE and SAE on two data sets, we noticed that LightGBM provides a more accurate estimation against the other two NN-based approaches in energy disaggregation of an individual appliance. With the largest values of SAE across all selected appliances, the GAN-based models generate the largest errors in estimating total energy consumption. CNN also shows worse performance on SAE than LightGBM. Regarding MAE, the GAN only shows good performance on the

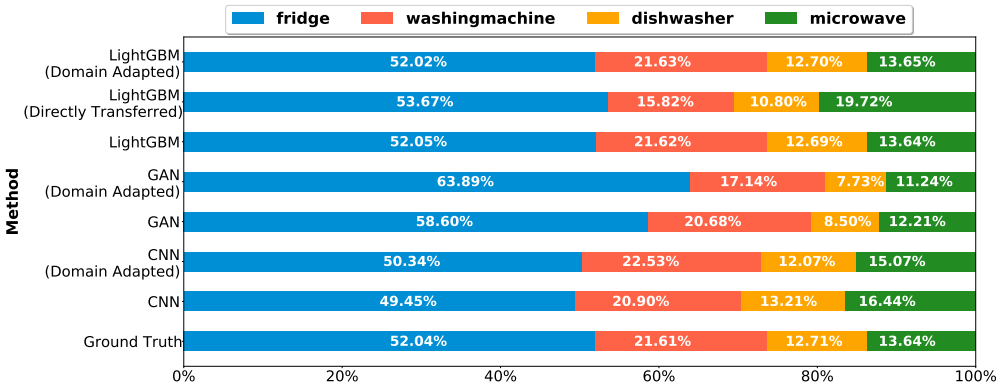


FIGURE 3.4: EEFI and AEFI on REDD

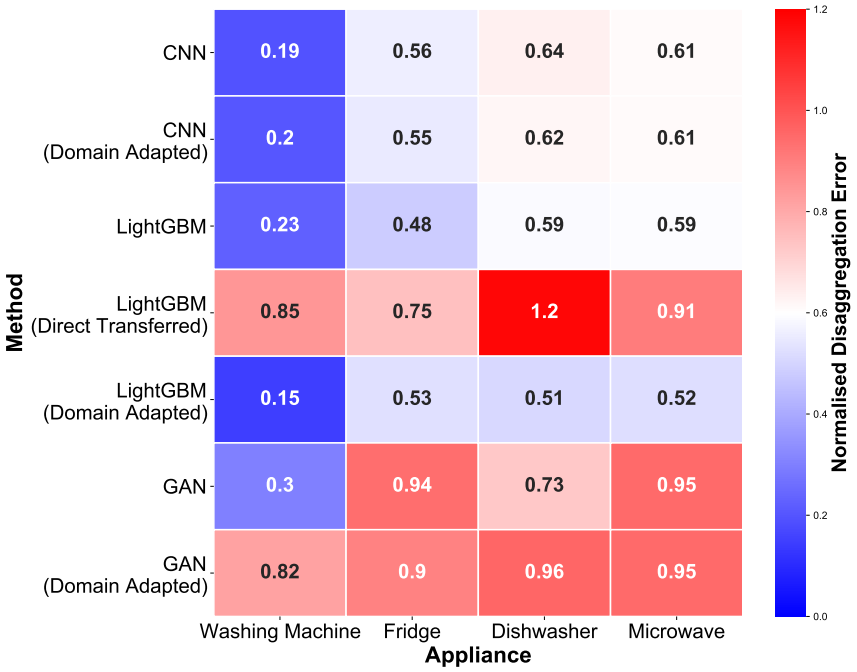


FIGURE 3.5: NDE Comparison on REDD

dishwasher and microwave but not the others. Especially for the fridge, the MAE value of GAN is 2× higher than that of LightGBM. Also, its NDE values are extremely high and vary a lot across appliances. These results indicate that GAN always incurs significant errors on the subset of tested instances and cannot provide stable effectiveness on different appliances. Instead, the NDE values for LightGBM and CNN are relatively small and merely vary within a limited range. Besides, the LightGBM model has the lowest E_{Acc} on both UK-DALE and REDD, showing the best overall performance on NILM applications. Figure 3.3 and 3.4 also

show the deviations between the estimated energy fractions of all tested models and actual energy fractions. Compared to CNN and GAN, the fractions of energy estimates produced by the LightGBM model are more close to the fractions of energy consumed by customers.

To investigate the cost-efficiency of three NILM approaches, we carefully examined the model size, training cost and inference cost, respectively. TABLE 3.5 shows the comparison result on UK-DALE. It indicates the storage and computation overhead of both CNN and GAN models is far higher than LightGBM. As shown in TABLE 3.5, the average model sizes of two neural networks reach 11935KB and 182896KB, 14× and 220× larger than LightGBM. As for running costs, the average training time of LightGBM goes 162.42s, while the average inference time is only 8.09s. LightGBM achieves 35× and 72× speedup over CNN and GAN in model training. The average inference time of LightGBM is 10× and 26× faster than two NN-based methods. The expensive computation and storage overhead will preclude NN-based models from a wide range of devices with limited resources. Overall, with joint consideration of accuracy and cost-efficiency, LightGBM provides superior performance against the other two neural networks for NILM applications.

3.4.5 Performance of Transfer Learning

From TABLE 3.2 and 3.3, we noted that the performance of LightGBM, CNN and CAN models for the same target appliance vary considerably between two data sets when three metrics are jointly considered. The results indicate that either user habits on appliances or signatures of appliances are significantly different in these two regions, leading to a difference in the accuracy of the established models for the same appliance. It also empirically verified that data drift occurred between REDD and UK-DALE, the same as a result from feature importance analysis in section 3.3.3. We thus then considered performing domain adaptation for the models trained from UK-DALE and then examined the performance of the domain-adapted models on REDD. This section validated the feasibility of our designed transfer learning approach for tree-based NILM models by comparing it with other techniques by the selected metrics. For LightGBM models, both direct sharing and our proposed algorithm were implemented and comprehensively evaluated. As for performance baselines, domain

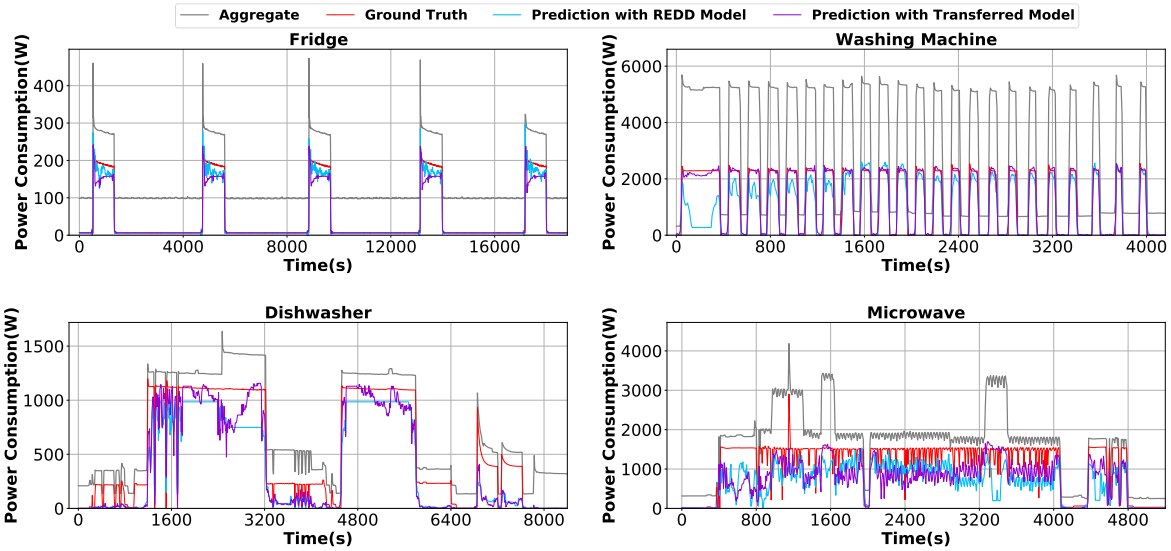


FIGURE 3.6: The power consumption estimations for the four appliances on REDD by two types of models: the standard LightGBM models trained on REDD, and the domain adapted LightGBM models trained on UK-DALE and fine-tuned on REDD

adaptation was also performed on CNN and GAN models by using transfer learning techniques in [34] and [5] accordingly.

Regarding the NN-based models, CNN and GAN show different experimental results in TABLE 3.4 when applying a similar fine-tuning technique. Compared with the models directly trained on REDD, the performance of the domain-adapted CNN models are remarkably improved. In contrast, GAN only increases its accuracy on the fridge model but not the others. As shown in Figure 3.5 and TABLE 3.4, domain-adapted CNN models, more or less, increase NDE and SAE of the corresponding directly-trained models. Interestingly, CNN outperforms GAN after domain adaptation is realised on the target data set.

For the direct transfer approach, LightGBM models for the four appliances are trained on UK-DALE individually, then used on REDD directly. As shown in TABLE 3.4, the errors of estimation from all appliance models dramatically increased and each model provides poor performance on all metrics. Additionally, in Figure 3.5, all the components in the row of direct transferred are red, which illustrate that this approach brings high normalised disaggregation errors to four appliances. For dishwasher, its NDE value is larger than 1 so that the model trained on UK-DALE failed to produce satisfactory estimates on the target domain.

We performed model-based domain adaptation when transferring the LightGBM models from source to target domain in our approach. To fine tune the parameters for each tree, it is of great importance to firstly restore the structure of NILM models constructed on UK-DALE. By using the LightGBM toolkit, we extracted a detailed model structure from the dumped LightGBM model. Based on the collected model structures, we successfully performed domain adaptation on the source models, and then the revised models were examined accordingly on REDD. As shown in TABLE 3.4, we can see that the domain-adapted LightGBM models by our approach outperform the other transferred models in all metrics. Unlike transfer learning on CNN and GAN, domain-adapted LightGBM lowers the MAE value without sacrificing the performance of the other metrics. Compared with LightGBM, CNN and GAN models trained on REDD, our approach still performs better in our experiment. Figure 3.5 also shows the high performance of our revised LightGBM model, as it always shows the lowest NDE value and increased stability for most appliances on REDD. Besides, we assess the overall performance of transferred models by comparing the long-period energy estimation performance, EEFI and AEFI, for the four selected appliances. The results are shown in Figure 3.4. Both LightGBM models, trained on REDD and the used domain-adaptation, provide a better approximation for the ground truth than other models, which resembles the result of E_{Acc}. To provide more intuitive performance comparison results, we plotted the energy estimates with a given period for each appliance as shown in Figure 3.6. It is clear that our transfer learning algorithm enables the increase in accuracy and provides near-ground-truth power estimation. For such performance improvement, we believe our design makes the target model inherit latent knowledge of both domains without increasing the complexity of the model structure and generalises the model established on the source domain. Thus, the transferred model has a better potential to perform well on unknown scenarios.

3.5 Summary

In this work, we design a cost-effective learning approach for energy disaggregation with low-frequency main readings. The first attempt is made to integrate LightGBM with the sequence-to-point training paradigm. Compared with the state-of-the-art techniques, LightGBM shows

remarkable gains in both accuracy and cost-efficiency. However, we cannot always train an oracle model when the availability of training data is limited. We thus proposed a unique model-based transfer learning method for tree-based ensemble models to reduce data collection overhead and extra training costs while maintaining comparable accuracy in a new target domain. As only the training samples from the target domain are required, the users' behaviours and the privacy of end-users from the source domain are reliably protected. In the experiments, by using our designed algorithm, our models show intact performance in the target domain and enhance the generability of the original models.

CHAPTER 4

Multi-source Domain Adaptive Network for Non-intrusive load Monitoring

---

Learning-based methods are the new trends in non-intrusive load monitoring (NILM) implementations but require large labelled data to work properly at end-user premises. In this chapter, we formulate an unsupervised multi-source domain adaptation problem to address this challenge by leveraging rich public datasets for building the NILM model. Next, we prove a new generalisation bound for the target domain under multi-source settings. A hybrid loss-driven multi-source domain adversarial network (HLD-MDAN) is developed by approximating and optimising the bound to tackle the domain shift between source and target domains. We conduct extensive experiments on three real-world residential energy datasets to evaluate the effectiveness of HLD-MDAN, showing that it is superior to other methods in single-source and multi-source learning scenarios.

4.1 Introduction

Chapter 3 presents an impactful and innovative source-free transfer learning approach to mitigate the concerns posed by data scarcity and domain drift within gradient boosting trees. For better use of state-of-the-art NILM techniques, this chapter seamlessly delves into the intricate landscape of deep learning and discusses the pressing challenges when neural networks are built for accurate energy disaggregation. First, the success of deep neural networks relies greatly on the availability of large-scale labelled datasets. The collection and labelling of the data from individual appliances are costly and error-prone as end-users often lack equipment and domain knowledge. The concern of privacy leakage also precludes users

from sharing sensitive appliance-level information and measurements. Second, public datasets with annotations are now becoming available for NILM studies. They can be combined to train models. However, these models may suffer from accuracy degradation when deployed to new users due to the distribution discrepancy between training and unseen activity samples. This phenomenon is known as domain shift [22] and stems from usage habits, appliance working patterns, and environmental noise. Third, domain adaptation (DA) is a powerful tool that is used to address domain shifts. To date, limited studies [69, 63] use DA in NILM algorithms and these feature the single-source DA approach. Using multiple sources can provide richer data but these sources may have different relationships to the target. Naively combining sources will incur a risk of negative transfer when the training involves sources irrelevant to the target. With multi-source data, NILM model training requires a domain-adaptive technique to extract statistical relationships among domains to rule out detrimental data sources.

This work is the first attempt to introduce multi-source DA to NILM. We design an adversarial multi-source learning approach to fully exploit multiple open-source labelled datasets and limited unlabelled target data for model construction. The insufficient data issue and domain shift are thus handled by transferring domain-invariant and task-specific knowledge across domains. Also, our approach attributes weights to each source according to domain discrepancy with the target domain. The source weights are optimised by model training to explore the statistical relationships between the target domain and source domains.

The main contributions of this work are threefold:

(1) We perform a theoretical analysis to prove a new generalisation bound for domain adaptation of the NILM models under a multi-source, single-target setting. The new bound is tighter than existing ones as it combines source risk, marginal shift and conditional shift to ensure successful domain adaptation in a regression task.
(2) With the new bound, we define a hybrid loss function to guide the optimisation of model parameters. A hybrid loss-driven multi-source domain adversarial network (HLD-MDAN) is proposed to realise domain adaptation for deep neural networks.

Also, the embedded weighting scheme enables the automatic selection of related houses during the model training.

(3) We comprehensively investigate our approach's adaptation in single-source and multi-source scenarios through conducting comparative experiments with other popular algorithms on real-world datasets.

4.2 Background

The core idea of DA is to ensure models from the source domain generalise well in the target domain. [12] discloses the underlying nature of domain adaptation by proving a generalisation bound for target risk, which is highly related to distribution discrepancy between domains in feature space. Some works utilise maximum mean discrepancy (MMD) [125], correlation alignment (CORAL) [113], and central moment discrepancy (CMD) [128] to realise the distribution match. [70] leverages multi-kernel MMD to optimise deep representation for domain alignment, and [71] extends MMD to match the joint distributions of multiple domain-specific layers across domains. Besides, adversarial learning [39, 64] performs well in extracting domain-invariant representations for DA. These approaches, however, are designed for single-source DA and often lead to negative transfer in a multi-source scenario. The main routine of multi-source DA is deriving optimal source weights for adversarial learning. In [136, 104], the authors propose different weighting schemes based on their theoretical analysis results. These works build on the covariate shift assumption but omit the conditional shift.

Deep learning is a recent trend in NILM studies, including convolutional neural networks (CNN) [130, 111], recurrent neural networks (RNN) [49], denoising auto-encoders [16] and generative adversarial networks [11]. Although these deep learning solutions all outperform traditional solutions, they often overfit the training data. We often encounter accuracy degradation when using a model trained in one dataset on another, due to the distribution discrepancy between the datasets. The classic domain adaptation techniques were adopted in [63, 69] to match the feature distribution over different datasets for NILM. However, these

works build on single-source DA methods, in which all the gathered energy datasets are directly combined into a unified source for the adaptation process. They neglect crucial findings that the relationship between any two domains is different [12]. Naive combining of datasets implies that equal weights are assigned to different data sources during model training. In addition, the approach used to reduce domain discrepancy between source and target domains is mainly designed for classification rather than regression, so it often fails to realise domain adaptation for NILM problems.

4.3 Problem Statement

In this section, we first analyse the statistical characteristics of energy consumption data from different domains and instantiate three domain shift modes that potentially impact the model's generalisation ability in practical application. Then, we mathematically model a multi-source domain adaptation problem for NILM.

4.3.1 Domain Shift Modes

The NILM models are built on aggregated consumption data from power meters embedded with informative knowledge of the target appliances' operational status. A major hypothesis of traditional learning approaches states that the training data (source domain) shares the same distribution with the test data (target domain). This hypothesis does not hold in NILM applications since the statistical distributions of different domains change along with environmental variations, namely domain shift [22].

Domain shift often stems from users' habits, appliance working patterns, and environmental noise. We verified this phenomenon by illustrating the distribution discrepancy of active power consumption of a target appliance and the whole-house load between households. The houses are randomly selected from REFIT (F), REDD (R), and UKDALE (U). We visualise the results in Figure 4.1 and 4.2. The number, followed by #, is the corresponding dataset's house number. The aggregated loads and the target appliance's power readings correspond to the features and labels in an energy disaggregation task. According to Figure 4.1, the

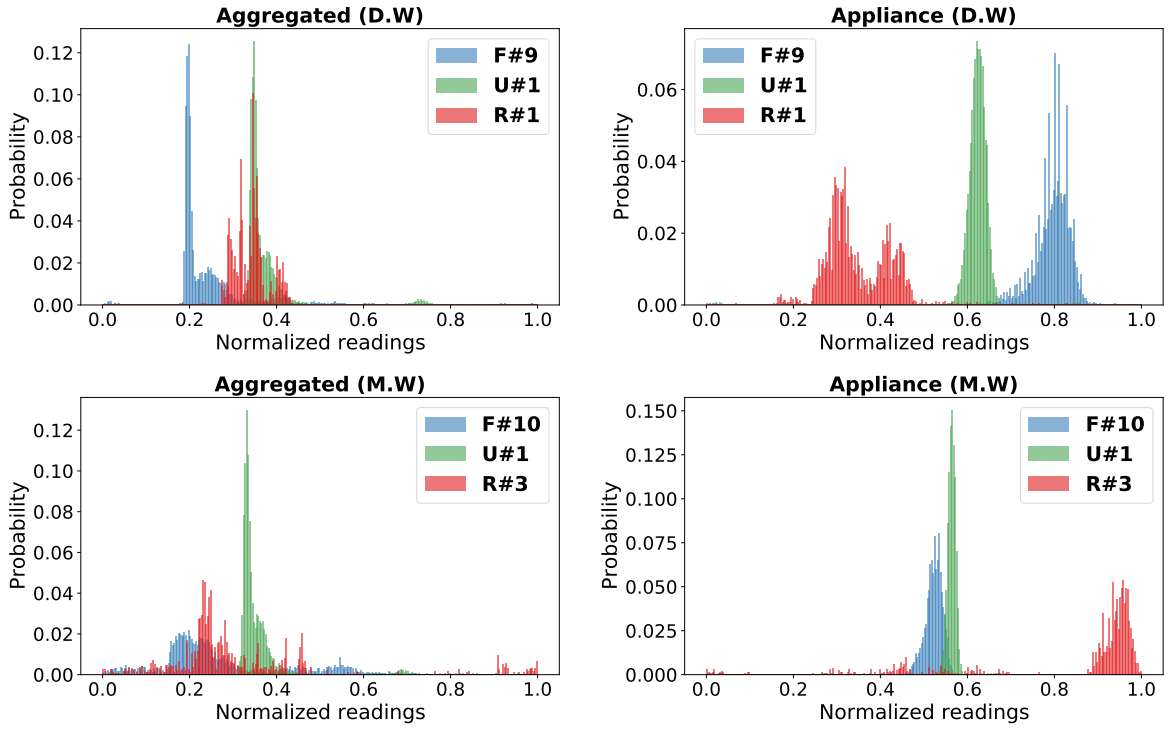


FIGURE 4.1: Distribution of normalised main readings and power consumption of active dishwasher and microwave over randomly selected houses from REFIT, UK-DALE, and REDD

distribution shape and the value range of both feature and label possibly change over houses from three different datasets. Figure 4.2 examines the distribution gaps between houses from the same dataset and indicates the same results. From the results, we identified three common domain shift modes:

- The shape of the feature distribution varies, e.g. F#10 and U#1 (M.W).
- The range of the feature values shifts without changing the distribution shape, e.g. F#9 and U#1 (D.W).
- The label distribution shifts while the feature distribution remains similar, e.g. F#4 and F#12 (M.W).

Domain shift could cause performance degradation and even make the trained models lose efficacy on the target domain. As a promising technique in the family of transfer learning, domain adaptation can be applied to enhance the transferability and generalisation ability of the trained NILM models.

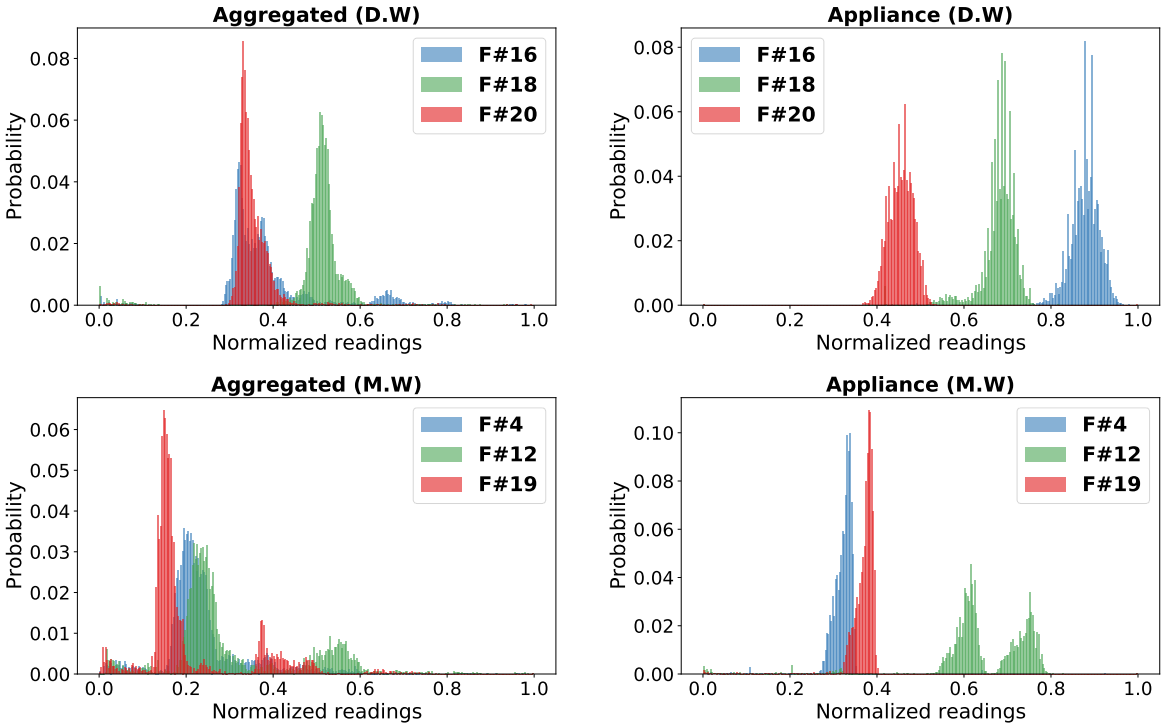


FIGURE 4.2: Distribution of normalised main readings and power consumption of active dishwasher and microwave across houses from REFIT

4.3.2 Multi-souce Domain Adaptation for NILM

The increasing availability of open-source labelled NILM datasets provides rich data for model training and validation. Using a single-source domain adaptation on the naive merged datasets is far from optimal. This strategy makes models suffer a high risk of negative transfer, theoretically and empirically verified in [136]. To this end, we propose a multi-source learning framework to incorporate data from diverse source domains into the target domain for NILM applications. Before introducing technical details, we first introduce the notations as Table 4.1 and define the problem of multi-source domain adaptation (MSDA) for NILM.

Suppose we have N independent labelled datasets to form different source domains. We define the i^{th} source domain as $< D_{S_i}, f_{S_i} >$, where D_{S_i} is the marginal distribution of input data X_{S_i}, and f_{S_i} is the true labelling function to map X_{S_i} to corresponding labels Y_{S_i}. In an energy disaggregation task, $x^j_{S_i} \in X_{S_i}$ and $y^j_{S_i} \in Y_{S_i}$, the feature and the label of an input instance represent the aggregated load consumption and the power usage of the target

TABLE 4.1: Key Notations

Notation	Definition
S_i	The i^{th} source domain
T	Target domain
D_{S_i}	The marginal distribution of input data from S_i
$\hat{D}_{S_i}$	The empirical distribution of sampled data from S_i
f_{S_i}	The labeling function over S_i
X_{S_i}	The feature set on S_i
Y_{S_i}	The label set on S_i
D_T	The marginal distribution of input data from T
$\hat{D}_T$	The empirical distribution of sampled data from T
f_T	The labeling function over T
X_T	The feature set on T
Y_T	The label set on T
$\mathcal{X}$	Feature space
$\mathcal{Y}$	Label space
$\mathcal{H}$	Hypothesis space
h	A hypothesis from $\mathcal{H}$ / The desired predictor
ℓ	The loss function defined over label space
$\epsilon_{S_i}(h)$	The source risk of S_i
$\epsilon_T(h)$	The target risk
$\hat{\epsilon}_{S_i}(h)$	The empirical source risk of S_i
$\hat{\epsilon}_T(h)$	The empirical target risk
$d^*_{\mathcal{H}}$	The domain discrepancy over a given hypothesis
α_i	The domain weight of S_i
L_e	The energy disaggregation loss
L_d	The domain discrepancy loss
L_c	The conditional shift loss
g	The feature extractor
h'_i	The domain discriminator for S_i
θ_g	The model parameters of g
θ_h	The model parameters of h
$\theta_{h'_i}$	The model parameters of h'_i

appliance, respectively. Similarly, the target domain is defined as $< D_T, f_T >$, but the labels Y_T cannot be given in the unsupervised setting. As we only consider the homogeneous transfer in this work, all different domains share the same feature space $\mathcal{X}$ and label space $\mathcal{Y}$. We also define a hypothesis h: $\mathcal{X} \to \mathcal{Y}$ of a hypothesis class $\mathcal{H}$ and a loss ℓ: $\mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$. The loss is supposed to be bounded by M over $\mathcal{Y}$. Given an instance x, the loss of a hypothesis $h \in \mathcal{H}$ w.r.t. the labelling function f is $\ell(h(x), f(x))$. For $\forall h, h' \in \mathcal{H}$, the average loss of two hypotheses under a distribution D is defined as: $\epsilon_D(h, h') = \mathbb{E}_{x \sim D}[\ell(h(x), h'(x))]$. It is a measure to estimate the distance of two hypotheses under a certain distribution. When

$h'(x) = f_{S_i}(x)$ is fixed, the source risk of domain D_{S_i} in terms of the hypothesis h can be defined as $\epsilon_{S_i}(h) = \epsilon_{S_i}(h, f_{S_i})$. We also use $\hat{\epsilon}_{S_i}(h)$ to denote the empirical risk of h over $\hat{D}_{S_i}$. $\hat{D}_{S_i}$ is the empirical distribution induced by m i.i.d. samples drawn from D_{S_i}. Similarly, $\epsilon_T(h)$ and $\hat{\epsilon}_T(h)$ represent the target risk and the corresponding empirical risk.

The objective of the multi-source NILM learning is to find an optimal hypothesis h that minimises the target risk $\epsilon_T(h)$. In practice, no labelled data is available in the target domain, and it is hard to directly approximate f_T using only the aggregated load readings. Based on the computational learning theory [7], the generalisation bound on $\epsilon_T(h)$ can be formulated by the convex combination of $\{\hat{\epsilon}_{S_i}(h)\}_{i=1}^N$ and the discrepancy between $\{\hat{D}_{S_i}\}_{i=1}^N$ and $\hat{D}_T$. Thus, the objective converts to minimise the generalisation bound on $\epsilon_T(h)$ by optimising model parameters. The theoretical details and proofs are provided in the following sections.

4.4 Theoretical Analysis for MSDA

This section defines the discrepancy of two distributions with the same hypothesis h for a regression task. We prove a new generalised bound of target risk for the MSDA problem using the domain discrepancy and the Rademacher complexity theory.

4.4.1 Technical Tools

First, three core lemmas are given for the following proof.

LEMMA 1 ([9]). *Let $\mathcal{H}$ be a hypothesis class mapping $\mathcal{X}$ to $[0, 1]$. Given a sample set $S = \{x_1, ..., x_m\}$ of size m drawn from the distribution D, for all $\delta \in (0, 1)$, with probability at least $1 - \delta$, the following inequality holds for all $h \in \mathcal{H}$:*

$$\mathbb{E}_{x\sim D}[h(x)] \leq \frac{1}{m}\sum_{i=1}^{m} h(x) + 2\hat{\mathcal{R}}_S(\mathcal{H}) + 3\sqrt{\log(2/\delta)/2m}$$

LEMMA 2 (Ledoux-Talagrand's contraction lemma [43]). *Let $\mathcal{H}$ be a hypothesis class over $\mathcal{X}$ and ϕ be a L-Lipschitz function. For samples S generated from any distribution, we have:*

$$\mathcal{R}_S(\phi \circ \mathcal{H}) \leq L\mathcal{R}_S(\mathcal{H})$$

where $\phi \circ \mathcal{H} := \{\phi \circ h \mid h \in \mathcal{H}\}$ is a class of composite functions.

LEMMA 3 ([77]). *Let $\mathcal{H}$ be a hypothesis class over $\mathcal{X}$, and $\ell : Y \times Y \to \mathbb{R}$ be a loss function bounded by $M > 0$. Let D be a distribution over $\mathcal{X}$ and $\hat{D}$ is its corresponding empirical distribution induced by m i.i.d. samples $S = \{x_1, ..., x_m\}$ drawn from D. Given a discrepancy distance $disc_\ell$ for D and $\hat{D}$ over ℓ, for all $\delta \in (0, 1)$, with probability at least $1 - \delta$, we have:*

$$disc_\ell(D, \hat{D}) \leq \hat{\mathcal{R}}_S(\mathcal{H}^*_\ell) + 3M\sqrt{\log(2/\delta)/2m}$$

where $\mathcal{H}^*_\ell := \{\ell(h(x), h'(x))|\ h, h' \in \mathcal{H}\}$.

4.4.2 Domain Discrepancy for Regression

Then, we introduce the basics of regression-based domain discrepancy (RBDD).

DEFINITION 4.4.1. Given a hypothesis class $\mathcal{H}$ on an instance space $\mathcal{X}$ and two distributions D and D' over $\mathcal{X}$, for any hypothesis $h \in \mathcal{H}$, the discrepancy between the two distributions w.r.t. h is defined as:

$$d^*_{\mathcal{H}}(D, D'; h) := \max_{h' \in \mathcal{H}} |\epsilon_D(h, h') - \epsilon_{D'}(h, h')| \tag{4.1}$$

RBDD reflects the discrepancy between two distributions as the difference between two regression losses. To measure this domain discrepancy, we search for another hypothesis $h' \in \mathcal{H}$ that maximises $|\epsilon_D(h, h') - \epsilon_{D'}(h, h')|$. In practice, h' is set as close to h on the source

domain as possible. The similarity between two distributions can thus be easily estimated with finite instances. If two distributions are not discriminative over $\mathcal{H}$, the value of $\epsilon_D(h, h')$ will be close to $\epsilon_{D'}(h, h')$. From the definition of (4.4.1), we have the below lemma to bound target risk.

LEMMA 4. *Let D and D' be two distributions over an instance space $\mathcal{X}$, for any $h, h' \in \mathcal{H}$,*

$$|\epsilon_D(h, h') - \epsilon_{D'}(h, h')| \leq d^*_{\mathcal{H}}(D, D'; h) \tag{4.2}$$

4.4.3 A Generalisation Upper Bound for Domain Adaptation

Before presenting the generalisation bound of target risk, there is a lemma introducing two triangle inequalities w.r.t. ϵ_D and $d^*_{\mathcal{H}}$, for the proofs of the following theorems.

LEMMA 5. *For any hypothesis space $\mathcal{H}$ and any distributions D, D', D^* over $\mathcal{X}$, if the loss function ℓ for regression problem follows triangular inequality, the following inequalities holds:*

$$\forall h_1, h_2, h_3 \in \mathcal{H}, \epsilon_D(h_1, h_2) \leq \epsilon_D(h_1, h_3) + \epsilon_D(h_3, h_2) \tag{4.3}$$

$$d^*_{\mathcal{H}}(D, D'; h) \leq d^*_{\mathcal{H}}(D, D^*; h) + d^*_{\mathcal{H}}(D^*, D'; h) \tag{4.4}$$

PROOF. In view of the definition of $\epsilon_D(h_1, h_2)$, we have:

$$\begin{aligned}
\epsilon_D(h_1, h_2) &= \mathbb{E}_{x\sim D}[\ell(h_1(x), h_2(x))] \\
&\leq \mathbb{E}_{x\sim D}[\ell(h_1(x), h_3(x)) + \ell(h_3(x), h_2(x))] \\
&= \epsilon_D(h_1, h_3) + \epsilon_D(h_3, h_2)
\end{aligned}$$

Similarly, it is easy to verify that the triangular inequality also holds on $d^*_{\mathcal{H}}(D, D'; h)$. For

any $h \in \mathcal{H}$, define $h' := \arg\max_{h' \in \mathcal{H}} |\epsilon_D(h, h') - \epsilon_{D'}(h, h')|$:

$$\begin{aligned} d^*_{\mathcal{H}}(D, D'; h) &= |\epsilon_D(h, h') - \epsilon_{D'}(h, h')| \\ &= |\epsilon_D(h, h') - \epsilon_{D^*}(h, h') + \epsilon_{D^*}(h, h') - \epsilon_{D'}(h, h')| \\ &\leq |\epsilon_D(h, h') - \epsilon_{D^*}(h, h')| + |\epsilon_{D^*}(h, h') - \epsilon_{D'}(h, h')| \\ &\leq d^*_{\mathcal{H}}(D, D^*; h) + d^*_{\mathcal{H}}(D^*, D'; h) \end{aligned}$$

where the first inequality comes from the triangular inequality of $|.|$, and the second one is due to Lemma 4. □

Combining $d^*_{\mathcal{H}}(D_S, D_T; h)$ and the source risk, we have the following theorem that characterises a generalisation bound on the target risk for domain adaptation.

THEOREM 4.4.1. *Let $\mathcal{H}$ be a hypothesis space over $\mathcal{X}$ and $< D_S, f_S >$,$< D_T, f_T >$ be source and target domains on $\mathcal{X}$. For any hypothesis $h \in \mathcal{H}$, the following bound related to target risk holds:*

$$\epsilon_T(h) \leq \epsilon_S(h) + d^*_{\mathcal{H}}(D_S, D_T; h) + \min\{\epsilon_S(f_S, f_T), \epsilon_T(f_S, f_T)\} \tag{4.5}$$

PROOF. The sketch of the proof is provided:

$$\begin{aligned} \epsilon_T(h) &\leq \epsilon_T(h, f_S) + \epsilon_T(f_S, f_T) + \epsilon_S(h, f_S) - \epsilon_S(h, f_S) \\ &\leq \epsilon_T(f_S, f_T) + \epsilon_S(h, f_S) + |\epsilon_T(h, f_S) - \epsilon_S(h, f_S)| \\ &\leq \epsilon_S(h) + d^*_{\mathcal{H}}(D_S, D_T; h) + \epsilon_T(f_S, f_T) \end{aligned}$$

where the first inequality is based on Lemma 5, the second one comes from the property of $|.|$, and the third one follows Lemma 4. By choosing to add and subtract $\epsilon_S(h, f_T)$ rather than $\epsilon_S(h, f_S)$ and deleting the use of triangle inequality in the first line, we also have:

$$\begin{aligned}\epsilon_T(h) &= \epsilon_T(h, f_T) + \epsilon_S(h, f_T) - \epsilon_S(h, f_T) \\ &\leq \epsilon_S(h, f_T) + |\epsilon_T(h, f_T) - \epsilon_S(h, f_T)| \\ &\leq \epsilon_S(h, f_T) + d^*_{\mathcal{H}}(D_S, D_T; h) \\ &\leq \epsilon_S(h) + d^*_{\mathcal{H}}(D_S, D_T; h) + \epsilon_S(f_S, f_T)\end{aligned}$$

where the property of $|.|$ and two lemmas are also applied to the deduction above. By combining these two inequalities, we obtain a tight bound as shown in Theorem 4.4.1. □

REMARK. This theorem explicitly indicates that three terms in (4.5) jointly determine the generalisation bound of target risk regarding a given hypothesis h. Similar to the bound in [136], the first term is the source risk, whereas the second denotes the marginal distribution discrepancy between the source and target domains. These two terms, however, are not good enough to ensure the success of domain adaptation in NILM applications. In our derived bound, the third term measures the discrepancy between the labeling functions of the two domains. It is another necessary condition for successful domain adaptation and reflects the underlying conditional shift problem, corresponding to the third shift mode previously mentioned.

We then extend the generalisation bound to the setting of MSDA for the regression problem as follows:

THEOREM 4.4.2. *Let $\mathcal{H}$ be a hypothesis class over $\mathcal{X}$. Let D_T and $\{D_{S_i}\}_{i=1}^N$ be the target distribution and N source distributions over $\mathcal{X}$. Given a vector α =$\{\alpha_1, ..., \alpha_N\}$ of domain weights for $\{D_{S_i}\}_{i=1}^N$, with $\sum_{i=1}^N \alpha_i = 1$ always satisfied, we have:*

$$\epsilon_T(h) \leq \sum_{i=1}^{N} \alpha_i \big(\epsilon_{S_i}(h) + d^*_{\mathcal{H}}(D_{S_i}, D_T; h)\big) + \min\{\epsilon_\alpha(f_\alpha, f_T), \epsilon_T(f_\alpha, f_T)\} \tag{4.6}$$

where ϵ_α is the source risk of the weighted combination domain D_α, and f_α is its labelling function.

PROOF. First, we give a definition of $\epsilon_\alpha(h, f_\alpha)$ and $d^*_{\mathcal{H}}(D_\alpha, D_T; h)$ to better support the proof. $\forall h \in \mathcal{H}$, define $\epsilon_\alpha(h, f_\alpha) := \sum_{i=1}^N \alpha_i \epsilon_{S_i}(h)$ and $d^*_{\mathcal{H}}(D_\alpha, D_T; h) := \sum_{i=1}^N \alpha_i d^*_{\mathcal{H}}(D_{S_i}, D_T; h)$. By replacing D_s with D_α and following the proof of Theorem 4.4.1, we have:

$$\begin{aligned}
\epsilon_T(h) &\leq \epsilon_T(h, f_\alpha) + \epsilon_T(f_\alpha, f_T) + \epsilon_\alpha(h, f_\alpha) - \epsilon_\alpha(h, f_\alpha) \\
&\leq \epsilon_T(f_\alpha, f_T) + \epsilon_\alpha(h, f_\alpha) + |\epsilon_T(h, f_\alpha) - \epsilon_\alpha(h, f_\alpha)| \\
&\leq \epsilon_\alpha(h) + d^*_{\mathcal{H}}(D_\alpha, D_T; h) + \epsilon_T(f_\alpha, f_T) \\
&= \sum_{i=1}^N \alpha_i \big(\epsilon_{S_i}(h) + d^*_{\mathcal{H}}(D_{S_i}, D_T; h)\big) + \epsilon_T(f_\alpha, f_T)
\end{aligned}$$

$$\begin{aligned}
\epsilon_T(h) &= \epsilon_T(h, f_T) + \epsilon_\alpha(h, f_T) - \epsilon_\alpha(h, f_T) \\
&\leq \epsilon_\alpha(h, f_T) + |\epsilon_T(h, f_T) - \epsilon_\alpha(h, f_T)| \\
&\leq \epsilon_\alpha(h, f_T) + d^*_{\mathcal{H}}(D_\alpha, D_T; h) \\
&\leq \epsilon_\alpha(h) + d^*_{\mathcal{H}}(D_\alpha, D_T; h) + \epsilon_\alpha(f_\alpha, f_T) \\
&= \sum_{i=1}^N \alpha_i \big(\epsilon_{S_i}(h) + d^*_{\mathcal{H}}(D_{S_i}, D_T; h)\big) + \epsilon_\alpha(f_\alpha, f_T)
\end{aligned}$$

By combining these two inequalities, we can obtain the bound in (4.6). □

REMARK. We introduce a weighted combination source D_α for domain adaptation. The vector of weights α shows the different relationships between the sources and the target and helps resist negative transfer. This bound can be reduced to (4.5) when having only one source domain.

With limited access to the samples from both source and target domains, the true statistical distributions of different domains are usually unknown. It is hard to calculate the true source risk and the discrepancy distance between source and target for a given source. Now, we bound $\epsilon_{S_i}(h)$ and $d^*_{\mathcal{H}}(D_{S_i}, D_T; h)$ by their corresponding empirical estimations, $\hat{\epsilon}_{S_i}(h)$ and $d^*_{\mathcal{H}}(\hat{D}_{S_i}, \hat{D}_T; h)$, which can be derived from finite training samples. We have the following

two lemmas to bound $\epsilon_{S_i}(h)$ and $d^*_{\mathcal{H}}(D_{S_i}, D_T; h)$ using the Rademacher complexity theory and its properties [9, 43, 77].

LEMMA 6. *Let $\mathcal{H}$ be a hypothesis class over $\mathcal{X}$, and ℓ be a L-Lipschitz loss bounded by M. For $\forall \delta \in (0,1)$ and a sample set $S = \{x_1, ..., x_m\}$ of size m drawn from the distribution D, with probability at least $1-\delta$, we have:*

$$\forall h \in \mathcal{H},\ \epsilon_S(h) \leq \hat{\epsilon}_S(h) + 2L\hat{\mathcal{R}}_S(\mathcal{H}) + 3M\sqrt{\log(2/\delta)/2m} \tag{4.7}$$

where $\hat{\mathcal{R}}_S(\mathcal{H})$ is empirical Rademacher complexity of $\mathcal{H}$ over S.

PROOF. We scale the loss ℓ to $[0,1]$ by dividing by M and thus obtain a new function class $\mathcal{H}_\ell/M$. According to the risk bound in terms of empirical Rademacher complexity presented in Lemma 1, we have:

$$\mathbb{E}_{x\sim D}[\ell(h(x), f_S(x))]/M \leq \frac{1}{m}\sum_{i=1}^{m} \ell(h(x_i), f_S(x_i))/M + 2\hat{\mathcal{R}}_S(\mathcal{H}_\ell/M) + 3\sqrt{\log(2/\delta)/2m}$$

For $\forall h \in \mathcal{H}$, we know $\epsilon_S(h) = \mathbb{E}_{x\sim D}[\ell(h(x), f_S(x))]$ and $\hat{\epsilon}_S(h) = \frac{1}{m}\sum_{i=1}^{m} \ell(h(x_i), f_S(x_i))$, where $\ell(h(x), f_S(x))$ is an element of the class of composite functions $\mathcal{H}_\ell := \{\ell(h(x), f_S(x))|\ h \in \mathcal{H}\}$. Following the property of empirical Rademacher complexity, we have $\hat{\mathcal{R}}_S(\mathcal{H}_\ell/M) = \frac{1}{M}\hat{\mathcal{R}}_S(\mathcal{H}_\ell)$. In regression problems, we often define loss $\ell(h(x), f_S(x)) := \phi(h(x) - f_S(x))$, where ϕ: $\mathbb{R} \to \mathbb{R}$ and L-Lipschitz. $\mathcal{H}_\ell$ also can be denoted as $\{\phi(h(x) - f_S(x))|\ h \in \mathcal{H}\}$. By Ledoux-Talagrand'scontraction lemma, we have $\hat{\mathcal{R}}_S(\mathcal{H}_\ell) \leq L\hat{\mathcal{R}}_S(\mathcal{H} - f_S)$ with $\mathcal{H} - f_S := \{h(x) - f_S(x)|\ h \in \mathcal{H}\}$. Due to the property of the empirical Rademacher complexity: for any fixed function f, $\hat{R}_S(\mathcal{H} + f) = \hat{R}_S(\mathcal{H})$, we have $\hat{R}_S(\mathcal{H} - f_S) = \hat{R}_S(\mathcal{H})$. Now we can conclude that $\hat{\mathcal{R}}_S(\mathcal{H}_\ell) \leq L\hat{R}_S(\mathcal{H})$. By combining all the above inequalities, the lemma is verified. □

LEMMA 7. *Let $\mathcal{H}$ be a hypothesis class over $\mathcal{X}$ and ℓ be a L-Lipschitz loss bounded by $M > 0$. For any two distributions D and D', let $\hat{D}$ and $\hat{D}'$ be the corresponding empirical distributions generated with m i.i.d. samples. For all $\delta \in (0,1)$, with probability at least*

$1-\delta$, *the below inequality holds for all* $h \in \mathcal{H}$*:*

$$d^*_{\mathcal{H}}(D, D'; h) \leq d^*_{\mathcal{H}}(\hat{D}, \hat{D}'; h) + 4L\hat{\mathcal{R}}_S(\mathcal{H}) + 6M\sqrt{\log(2/\delta)/2m} \tag{4.8}$$

PROOF. By Lemma 2 related to $d^*_{\mathcal{H}}(D, D'; h)$, we have:

$$d^*_{\mathcal{H}}(D, D'; h) \leq d^*_{\mathcal{H}}(D, \hat{D}; h) + d^*_{\mathcal{H}}(\hat{D}, \hat{D}'; h) + d^*_{\mathcal{H}}(\hat{D}', D'; h)$$

We define a class of functions $\mathcal{H}^*_\ell := \{\ell(h(x), h'(x))|\forall h, h' \in \mathcal{H}\}$. By Lemma 3, with probability at least $1-\delta$, we have:

$$d^*_{\mathcal{H}}(D, \hat{D}; h) \leq \hat{\mathcal{R}}_S(\mathcal{H}^*_\ell) + 3M\sqrt{\log(2/\delta)/2m}$$

Similarly, with probability at least $1-\delta$, we also have:

$$d^*_{\mathcal{H}}(D', \hat{D}'; h) \leq \hat{\mathcal{R}}_S(\mathcal{H}^*_\ell) + 3M\sqrt{\log(2/\delta)/2m}$$

$\forall h, h' \in \mathcal{H}$, we define loss $\ell(h(x), h'(x)) := \phi(h(x) - h'(x))$, where ϕ: $\mathbb{R} \to \mathbb{R}$ and L-Lipschitz. By Ledoux-Talagrand's contraction lemma, we have $\hat{\mathcal{R}}_S(\mathcal{H}^*_\ell) \leq L\hat{\mathcal{R}}_S(\mathcal{H}^*)$ with $\mathcal{H}^* := \{h(x) - h'(x)|\ h, h' \in \mathcal{H}\}$. With the definition of the Rademacher variables, $\mathcal{H}^*$ can be bounded by:

$$\begin{aligned}
\hat{\mathcal{R}}_S(\mathcal{H}^*) &= \mathbb{E}_\sigma[\sup_{h,h'} \frac{1}{m}|\sum_{i=1}^{m} \sigma_i(h(x_i) - h'(x_i))|] \\
&\leq \mathbb{E}_\sigma[\sup_{h} \frac{1}{m}|\sum_{i=1}^{m} \sigma_i(h(x_i))|] + \mathbb{E}_\sigma[\sup_{h'} \frac{1}{m}|\sum_{i=1}^{m} \sigma_i(h'(x_i))|] \\
&= 2\hat{\mathcal{R}}_S(\mathcal{H})
\end{aligned}$$

Hence, we can conclude that $\hat{\mathcal{R}}_S(\mathcal{H}^*_\ell) \leq 2L\hat{\mathcal{R}}_S(\mathcal{H})$. Using a union bound to combine all the inequalities above, we can get the bound as (4.8). □

With these two lemmas, we derive a new generalisation bound for MSDA in terms of $\hat{\epsilon}_{S_i}(h)$ and $d^*_{\mathcal{H}}(\hat{D}_{S_i}, \hat{D}_T; h)$ as below:

THEOREM 4.4.3. *Let $\mathcal{H}$ be a hypothesis class over $\mathcal{X}$. Let D_T and $\{D_{S_i}\}_{i=1}^N$ be the target distribution and N source distributions over $\mathcal{X}$. $\{\hat{D}_{S_i}\}_{i=1}^N$ and $\hat{D}_T$ are the corresponding empirical distributions generated with m i.i.d. samples from each domain. Given a vector α =$\{\alpha_1, ..., \alpha_N\}$ of domain weights, with $\sum_{i=1}^N \alpha_j = 1$ always satisfied, for all $\delta \in (0, 1)$, with probability at least $1 - \delta$, the following bound holds:*

$$\begin{aligned}\epsilon_T(h) \leq \sum_{i=1}^{N} \alpha_i \big(\hat{\epsilon}_{S_i}(h) + d^*_{\mathcal{H}}(\hat{D}_{S_i}, \hat{D}_T; h)\big) + \min\{\epsilon_\alpha(f_\alpha, \\ f_T), \epsilon_T(f_\alpha, f_T)\} + 6L\hat{\mathcal{R}}_{S_\alpha}(\mathcal{H}) + O(\sqrt{\log(1/\delta)/m})\end{aligned} \tag{4.9}$$

where $\hat{\mathcal{R}}_{S_\alpha}(\mathcal{H}) = \sum_{i=1}^N \alpha_i \hat{\mathcal{R}}_{S_i}(\mathcal{H})$.

REMARK. The upper bound of target risk in this theorem depends on four terms. The first term is a weighted combination of empirical source risks and RBDDs. The second term implicitly shows the distance of label functions between the target domain and the mixture source domain. The last two terms illustrate the other two factors related to the size of the hypothesis class and the number of sample instances.

4.5 Methodology

This section presents a neural network-based solution to the unsupervised MSDA for energy disaggregation. Using the generalisation bound in Theorem 4.4.3, we first detail the formulation of a hybrid loss, jointly considering regression loss on source domains, marginal shift, and conditional shift between source domains and target domain. The loss combination determines the model parameters and the feature representations as an optimisation objective. We then specify the implementation of an optimisation algorithm for model establishment.

4.5.1 Hybrid Losses for Domain Adaptation

Suppose we have m unlabelled instances from the target domain D_T and N labelled sources $\{D_{S_i}\}_{i=1}^N$ that each has labelled instances of size m. We introduce a representation space

Z and a feature extractor g: $\mathcal{X} \to Z$. g is parameterised by θ_g and achieves feature transformation of $\mathcal{X}$. A hypothesis from $\mathcal{H}$ is used as a predictor $h : Z \to \mathcal{Y}$. $h \circ g$ denotes the composite function $h(g(\cdot))$. To realise MSDA for NILM with collected instance data, we need to minimise the generalisation bound derived from Theorem 4.4.3. When data size and hypothesis class $\mathcal{H}$ are fixed, only three terms determine the bound value, i.e., the weighted combination of source risks, the marginal distribution discrepancy, and the conditional distribution discrepancy between the target domain and the N source domains. Considering these terms, we carefully design the following three losses to guide the neural network training jointly.

Energy Disaggregation Loss: We want to guarantee that the learned feature representations are task discriminative by minimising the weighted source risks. Without such constraint, the model will learn meaningless knowledge and perform energy disaggregation poorly. The energy disaggregation loss can be calculated by:

$$L_e = \sum_{i=1}^{N} \alpha_i \big(\frac{1}{m} \sum_{j=1}^{m} \ell(h(g(x_i^j)), y_i^j) \big) \tag{4.10}$$

where ℓ is a loss function following triangle inequality and symmetric property, and y_i^j is the true label of j^{th} instance from D_{S_i} domain.

Domain Discrepancy Loss: As discussed, small RBDD enables successful domain adaptation and makes it possible to generalise the trained neural network to the target domain. Based on Definition 4.4.1, we formulate the domain discrepancy loss as a weighted sum of RBDDs.

$$\begin{aligned} L_d &= \sum_{i=1}^{N} \alpha_i \max_{h_i' \in \mathcal{H}} |\epsilon_{S_i}(h \circ g, h_i' \circ g) - \epsilon_T(h \circ g, h_i' \circ g)| \\ &= \sum_{i=1}^{N} \alpha_i \max_{h_i' \in \mathcal{H}} |\frac{1}{m} \sum_{j=1}^{m} \ell\big(h(g(x_i^j)), h_i'(g(x_i^j))\big) - \frac{1}{m} \sum_{j=1}^{m} \ell\big(h(g(x_T^j)), h_i'(g(x_T^j))\big)| \end{aligned} \tag{4.11}$$

where h_i' is another trained predictor that assists in approximating the regression-based domain discrepancy between the source domain D_{S_i} and the target domain. In practice, we realise domain adaptation by simultaneously optimising g and h_i' to minimise the RBDD of two

distributions. To reach this goal, we reformulate the optimisation of L_d as a minimax saddle point problem:

$$\min_{\theta_g} \sum_{i=1}^{N} \alpha_i \max_{h_i' \in \mathcal{H}} |\epsilon_{S_i}(h \circ g, h_i' \circ g) - \epsilon_T(h \circ g, h_i' \circ g)| \tag{4.12}$$

Conditional Shift Loss: In Theorem 4.4.3, the second term of the bound implicitly reflects the discrepancy of the conditional distribution, and is another necessary condition for successful domain adaptation. To minimise the generalisation bound on target risk, we define a new loss with a distance measure, namely conditional embedding discrepancy [112], and align the conditional distributions by optimising the loss in the training process. As conditional embedding discrepancy is defined in terms of the conditional embedding operator, we first introduce the basics of such an operator. In this chapter, representations $Z \subseteq \mathcal{Z}$, generated from feature extractor g, and $Y \subseteq \mathcal{Y}$ are the inputs and outputs of the predictor h. $P(Y|Z)$ denotes the conditional distribution. We now embed the random variables Z and Y into corresponding reproducing kernel Hilbert spaces (RKHS) $\mathcal{H}_z$ and $\mathcal{H}_y$ by using feature maps $\psi : \mathcal{Z} \to \mathcal{H}_z$ and $\phi : \mathcal{Y} \to \mathcal{H}_y$. According to [112], the conditional embeddings of entire distribution $P(Y|Z)$ can be defined as:

$$C_{Y|Z} = C_{YZ} C_{ZZ}^{-1} = \mathbb{E}_{YZ}[\psi(Y) \otimes \phi(Z)] \mathbb{E}_{ZZ}^{-1}[\phi(Z) \otimes \phi(Z)] \tag{4.13}$$

where C_{YZ} is the cross-covariance operator defined in [8], C_{YZ} is the self-covariance operator, and $\otimes$ is the tensor product. The empirical estimate of $C_{Y|Z}$ on the source domain S_i and the target domain is:

$$\begin{aligned} \hat{C}_{Y|Z}^{S_i} &= \frac{1}{m} \Upsilon_{S_i} H_{S_i} \Phi_{S_i}^{\intercal} \big(\frac{1}{m} \Upsilon_{S_i} H_{S_i} \Upsilon_{S_i}^{\intercal}\big)^{-1} \\ \hat{C}_{Y|Z}^{T} &= \frac{1}{m} \Upsilon_{T} H_{T} \Phi_{T}^{\intercal} \big(\frac{1}{m} \Upsilon_{T} H_{T} \Upsilon_{T}^{\intercal}\big)^{-1} \end{aligned} \tag{4.14}$$

where $\Upsilon = (\psi(g(x_1)), ..., \psi(g(x_k)))$, $\Phi = (\phi(\hat{y}_1), ..., \phi(\hat{y}_k))$, $\hat{y}$ represents the output of the model, m is instance number for a given domain, and H denotes an idempotent centering matrix defined by $H = I - \frac{1}{m}\mathbf{1}\mathbf{1}^{\intercal}$. As in [67], the conditional embedding discrepancy is formulated as a Hilbert-Schmidt norm of two empirical conditional embeddings. We define

the conditional shift loss L_c for MSDA as a weighted combination of conditional embedding discrepancies.

$$L_c = \sum_{i=1}^{N} \alpha_i \left\| \hat{C}_{Y|Z}^{S_i} - \hat{C}_{Y|Z}^{T} \right\|_{\mathcal{H}_z \otimes \mathcal{H}_Y}^2 \tag{4.15}$$

In the model training, each weighted component of L_c can be formulated in terms of Gaussian kernels for $\mathcal{H}_z$ and $\mathcal{H}_y$, and [67] details the derivation process.

The core of domain adaptation is to construct a fine-grained model that allows domain-invariant feature representations by eliminating distribution shift while guaranteeing our desired task's success. According to the specification of the three network losses introduced above, we know that L_d and L_c cooperatively align the marginal and conditional distribution between the source domains and the target domain, and L_e contributes to the performance satisfaction of the learning task. To realise unsupervised MSDA, we define a hybrid loss function by combining three losses. The hybrid loss is formulated as:

$$L_{hybrid} = (1 - \lambda)L_e + \lambda[\mu L_c + (1 - \mu)L_d] \tag{4.16}$$

where λ and μ are the weight factors within $[0, 1]$. They jointly control the training attention over three losses and regularise their value scales. At the initial stage of model training, we let L_e dominate L_{hybrid} for better convergence and accuracy by setting λ to zero or to an extremely small value. When showing good overall performance on source domains, the model will concentrate on improving transferability on the target domain by dynamically changing λ value. To enable a better accuracy-transferability trade-off, the Adam optimisation strategy is applied for adaptively updating λ and μ.

4.5.2 Adversarial Learning for Parameters Optimisation

Inspired by Theorem 4.4.3, we propose HLD-MDAN to fulfil the goal of MSDA by minimising the generalisation bound. In practical applications, the objective is to optimise model parameters that realise the minimisation of hybrid loss. Fig.4.3 shows the general network

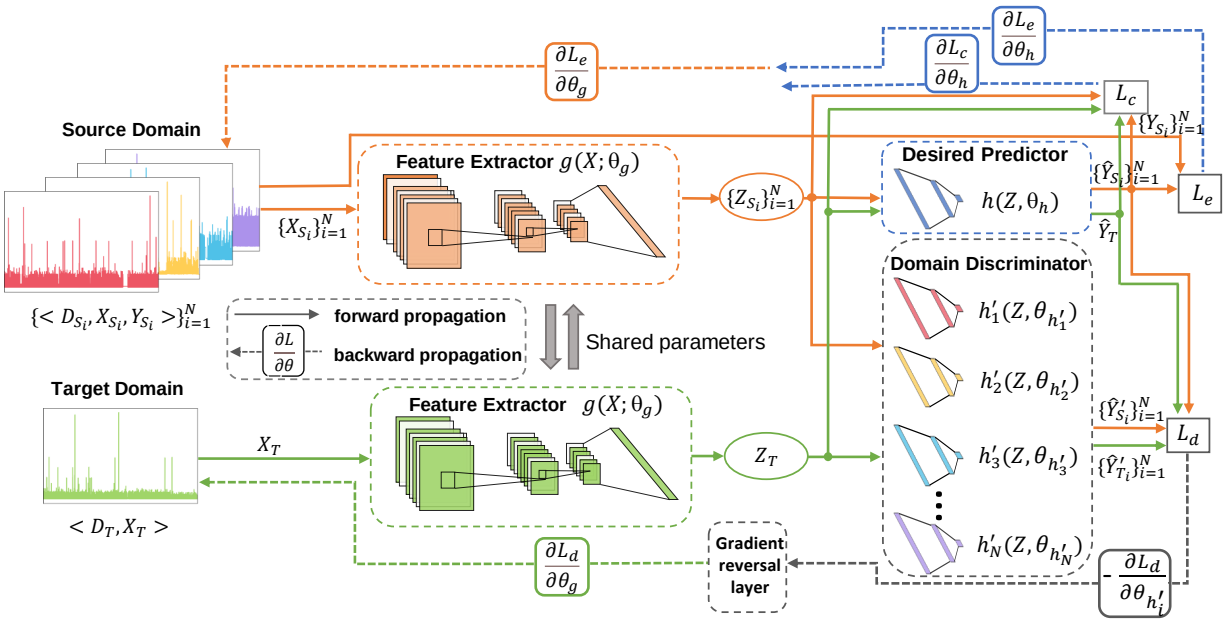


FIGURE 4.3: An overview of HLD-MDAN architecture

architecture and the implementation for MSDA. Three critical components, including feature extractor g, desired predictor h, and domain discriminators $\{h'_i\}_{i=1}^N$, constitute the whole neural network. θ_g, θ_h and $\{\theta_{h'_i}\}_{i=1}^N$ are their corresponding model parameters. Both g and h follow the interpretation in 4.5.1. g is used to extract high-level representations from inputs, sharing between source and target domains. All network structures with powerful feature representation capability can be selected as feature extractors. Convolutional neural networks (CNNs), recurrent neural networks (RNNs), and attention-based networks are widely used in the NILM problem. h is the desired predictor and is responsible for outputting the power consumption of the target appliance with computation results of g. We aim to generalise g and h well on the target domain by fully exploiting statistical relationships between labeled source data and unlabeled target data. Also, we develop a domain discriminator h'_i for each source domain S_i to approximate domain discrepancy loss L_d according to (4.11). h'_i shares the same structure with h but facilitates different model parameters. Since optimising L_d is a minimax problem, as presented in (4.12), we apply an adversarial training method to reduce domain shift and learn domain-invariant representations. In the training process, θ_g is trained to lower L_d, whereas $\theta_{h'_i}$ is updated to increase L_d for easier domain discrimination. They compete against each other to achieve distribution alignment between the source domains and

Algorithm 5 Implementation of HLD-MDAN

Input: The labeled datasets of N source domains $\{(X_{S_i}, Y_{S_i})\}_{i=1}^{N}$, the unlabeled dataset of target domain X_T, the dataset size m, pre-trained epochs M', maximum training epochs M, batch size n, weight parameter λ, μ

Output: The optimised parameters θ_g, θ_h, $\{\theta_{h_i'}\}_{i=1}^{N}$, α

Initialise parameters θ_g, θ_h and $\{\theta_{h_i'}\}_{i=1}^{N}$, η_g, η_h,$\{\eta_{h_i'}\}_{i=1}^{N}$ for corresponding model parts. Initialise domain weights $\alpha_i \in \alpha$ with $\alpha_i \leftarrow \frac{1}{N}$ and learning rate η_α
for $t = 1 \rightarrow M$ **do**
  for each minibatch $j = 1 \rightarrow m/n$ **do**
    Forward propagation
    Sample n instances from each domain as a minibatch
    $\{(X_{S_i}^{(j)}, Y_{S_i}^{(j)})\}_{i=1}^{N}$, $X_T^{(j)} \leftarrow Sample(\{(X_{S_i}, Y_{S_i})\}_{i=1}^{N}, X_T)$
    for $i = 1 \rightarrow$ N **do**
      $Z_{S_i}^{(j)} \leftarrow g(X_{S_i}^{(j)})$, $\hat{Y}_{S_i}^{(j)} \leftarrow h(Z_{S_i}^{(j)})$ *#Source domain*
    end for
    $Z_T^{(j)} \leftarrow g(X_T^{(j)})$, $\hat{Y}_T^{(j)} \leftarrow h(Z_T^{(j)})$ *#Target domain*
    for $i = 1 \rightarrow$ N **do**
      $\hat{Y}_{S_i}^{'(j)} \leftarrow h_i'(Z_{S_i}^{(j)})$, $\hat{Y}_{T_i}^{'(j)} \leftarrow h_i'(Z_T^{(j)})$ *#Discriminator*
    end for
    Compute L_e, L_c, L_d based on the outputs above
    $\lambda \leftarrow 0$ if $t < M'$
    $L_{hybrid} \leftarrow (1-\lambda)L_e + \lambda[\mu L_c + (1-\mu)L_d]$
    Backward propagation
    $\theta_g^{(t+1)} \leftarrow \theta_g^{(t)} - \eta_g(\bigtriangledown_{\theta_g} L_{hybrid})$
    $\theta_h^{(t+1)} \leftarrow \theta_h^{(t)} - \eta_h(\bigtriangledown_{\theta_h} L_e + \bigtriangledown_{\theta_h} L_c)$
    for $i = 1 \rightarrow$ N **do**
      $\theta_{h_i'}^{(t+1)} \leftarrow \theta_{h_i'}^{(t)} - \eta_{h_i'}(\bigtriangledown_{\theta_{h_i'}}(-L_d))$
    end for
    $\alpha^{(t+1)} \leftarrow \alpha^{(t)} - \eta_h(\bigtriangledown_\alpha L_d + \bigtriangledown_\alpha L_c)$
    $\alpha_i^{(t+1)} \leftarrow \alpha_i^{(t+1)} / ||\alpha^{(t+1)}||_1$ *#Normalize* $\alpha_i \in \alpha$
  end for
end for
return θ_g, θ_h, $\{\theta_{h_i'}\}_{i=1}^{N}$, α

the target domain. Similar to other adversarial domain adaptations [39], the gradient reversal layer is implemented in our architecture to optimise θ_g and $\theta_{h_i'}$ simultaneously in backward propagation. Besides, θ_h and α are also optimised by the related gradients. In our approach, the gradient of θ_h corresponds to L_e and L_c, while the convergence of α is determined by L_d and L_c. All different parameters are iteratively updated until the stop criteria are met. The pseudo-code of our solution is summarised in Algorithm 5.

4.6 Experiments

In this section, we tested our solution on real-world NILM datasets and compared it with the benchmark algorithms for performance evaluation.

4.6.1 Experimental Setup

Datasets: We used three publicly available datasets, REDD [54], UK-DALE [52], and REFIT [87], to showcase the adaptation performance of our method under unsupervised settings. The fridge (FG), washing machine (WM), dishwasher (DW), and microwave (MV) were selected as target appliances for energy disaggregation. Since the raw measurements in the three datasets were sampled differently, we first preprocessed data to unify the sampling rates every 8 seconds for both aggregated readings and appliance-level power consumption data. We adopted the sequence-to-point learning paradigm to generate input samples in the experiments. A fixed-size window n slides throughout a long-term sequence of measurements with the step size d. For a given start time point t of the window, a short sequence of aggregated readings $\{x_t,...,x_{t+n-1}\}$ is extracted as the feature vector. The corresponding label is the target appliance's power consumption at the window midpoint, denoted as $y_{t+n/2}$. We set the window size n as 19 for short-term energy disaggregation, and the step size d for REDD, UK-DALE, and REFIT as 2, 16, and 16, respectively.

Benchmarks: We selected four methods to be the benchmark algorithms for performance comparison. i) Baseline is a basic model trained without domain adaptation. ii) DANN+JMMD [69] is an adversarial neural network conditioned on the joint distribution loss for domain adaptation. It has been empirically proven to alleviate domain shifts in NILM. iii) MDAN [135] achieves MSDA by using $d_{\mathcal{H}}$ distance and an adversarial learning strategy similar to DANN. We adopted the soft version in this chapter. iv) AHD-MSDA [104] is an unsupervised DA method for regression tasks. We applied the network structure in [130] for all methods.

TABLE 4.2: Houses selected for evaluation of single-source and multi-source domain adaptation

App.	Single-source	Multi-source
FG	[F#15 **R#3**]	[R#1 **R#2 U#1** U#2 F#12 **F#15**]
WM	[F#16 **R#3**]	[**R#1** R#2 **U#1** U#2 F#9 **F#16**]
DW	[F#18 **R#1**]	[**R#1** R#2 **U#1** U#2 **F#13** F#18]
MV	[F#17 **R#1**]	[**R#1** R#2 **U#1** U#2 F#10 **F#17**]

Experiment implementation: The basic neural network comprises five convolutional layers as the feature extractor and one fully-connected layer as the desired predictor. For the feature extractor, we set the kernel size as 10, 8, 6, 5, and 5. The filter numbers are 30, 30, 40, 40, and 50, respectively. The input of the predictor is 1024. For model training, the learning rate of the Adam optimiser is 0.0001, and the maximum epoch number is 50 with a batch size of 64. To improve convergence speed, except for the baseline, the models constructed in other approaches were first pre-trained for 10 epochs only using the source data. Then domain adaptation-related loss started to work. All models were trained on a desktop computer with Intel i7 8700 CPU @2.81GHz, 32GB DDR4 RAM, and NVIDIA GTX 1080 Graphics Card with 8GB GDDR5X VRAM. The software packages used for the implementation include Python 3.7 and PyTorch 1.10.0 with CUDA 11.3.

4.6.2 Performance Comparison

This chapter investigated HLD-MDAN in single-source and multi-source scenarios using three widely-used metrics for NILM as introduced in Chapter 3, which include MAE, SAE, and NDE.

Single-source: Under the single-source setting, we randomly picked one labelled house from REFIT as the source domain and one house from REDD as the target domain for each target appliance. The details of house selection are provided in Table 4.2, with the bold entries representing target houses. We report the comparison results of single-source domain adaptation across different methods in the left part of Table 4.3, denoted as F$\rightarrow$R. HLD-MDAN outperformed benchmarks when three metrics were jointly considered, even though it was designed for multi-source learning problems. HLD-MDAN can significantly

TABLE 4.3: Results of domain adaptation with different learning approaches

App.	Methods	F → R			H_1^* → R			H_2^* → U			H_3^* → F		
		MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE
FG	i)	48.30	0.55	0.81	55.61	0.54	0.67	47.52	0.80	0.90	39.32	1.08	1.07
	ii)	46.80	0.53	0.80	50.84	0.51	0.65	41.66	0.55	0.83	33.81	0.55	1.02
	iii)	47.22	0.42	0.80	52.84	0.53	0.67	49.60	1.00	1.03	33.54	0.46	1.03
	iv)	46.28	0.49	0.81	57.43	0.64	0.74	42.82	**0.07**	**0.82**	38.30	0.70	1.06
	Ours	**44.47**	**0.37**	**0.78**	**42.90**	**0.36**	**0.57**	**40.52**	0.38	0.90	**31.84**	**0.43**	**0.95**
	Improved	7.93%	32.73%	3.70%	22.86%	33.33%	14.93%	14.73%	91.25%	8.89%	19.02%	60.19%	11.21%
WM	i)	41.52	0.73	0.99	34.65	0.86	0.96	56.96	1.82	0.96	81.21	2.81	1.10
	ii)	37.98	0.72	0.88	30.54	0.53	0.89	29.79	0.68	0.87	35.82	0.32	0.92
	iii)	37.84	0.70	0.98	37.46	0.53	0.87	37.44	0.83	0.91	65.96	2.17	1.00
	iv)	38.93	0.64	0.96	33.03	0.55	0.81	30.15	1.00	1.00	33.76	0.34	0.93
	Ours	**36.67**	**0.60**	**0.93**	**29.55**	**0.03**	**0.79**	**27.61**	**0.23**	**0.82**	**28.00**	**0.31**	**0.92**
	Improved	11.68%	17.81%	6.06%	14.72%	96.51%	17.71%	51.53%	87.36%	14.58%	65.52%	88.97%	16.36%
DW	i)	57.74	0.47	2.00	45.17	1.70	1.35	52.46	0.58	0.91	63.29	0.40	1.13
	ii)	39.79	**0.03**	1.13	42.16	**0.10**	1.38	30.60	0.41	0.77	59.34	1.65	1.20
	iii)	43.45	0.23	1.49	51.16	0.87	1.51	43.82	**0.20**	0.82	80.04	2.05	1.11
	iv)	38.49	0.16	1.18	39.79	0.45	1.29	34.08	0.59	1.00	61.16	2.10	**1.08**
	Ours	**31.76**	0.41	**1.13**	**30.39**	0.45	**0.97**	**21.54**	0.29	**0.76**	**52.31**	**0.15**	1.12
	Improved	44.99%	93.62%	43.50%	32.72%	94.12%	28.15%	58.94%	65.52%	16.48%	17.35%	62.50%	4.42%
MV	i)	44.59	0.08	1.00	32.60	0.53	0.86	40.95	0.60	0.90	54.98	1.64	1.14
	ii)	34.36	0.10	1.05	27.75	0.39	**0.81**	29.07	0.40	0.90	47.06	0.39	0.99
	iii)	42.98	0.15	1.05	36.38	0.73	0.92	48.40	**0.14**	0.83	64.05	1.73	1.15
	iv)	40.18	0.43	1.00	36.00	0.62	0.94	29.39	0.46	0.95	45.70	0.37	0.90
	Ours	**32.73**	**0.02**	**0.99**	**20.58**	**0.34**	0.89	**29.01**	0.40	**0.81**	**42.42**	**0.35**	**0.90**
	Improved	26.60%	75.00%	1.00%	36.87%	35.85%	5.81%	29.16%	76.67%	10.00%	22.84%	78.66%	21.05%

The row improvement shows the relative improvements in the best result of all methods compared with the baseline.

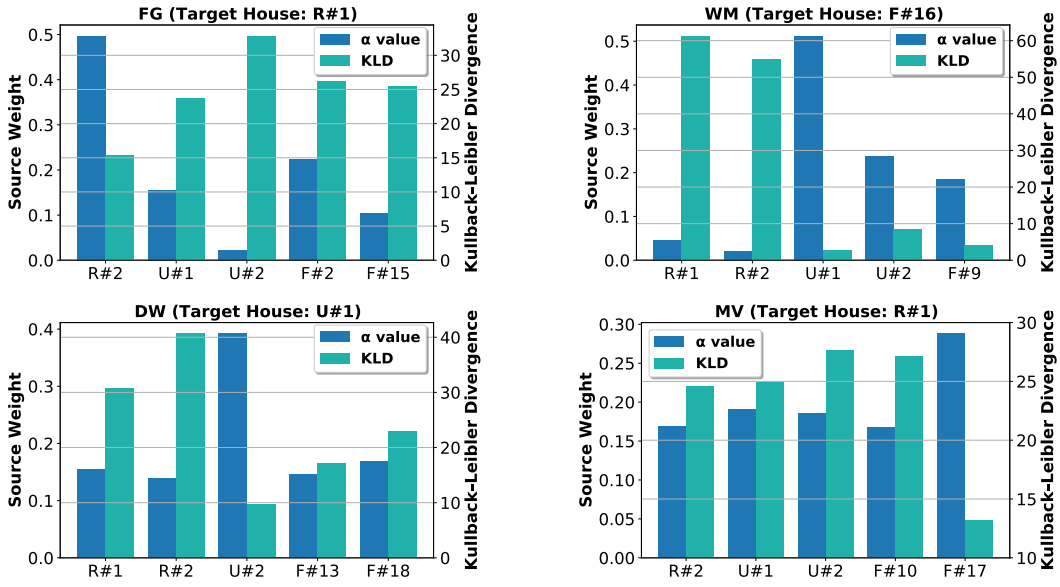


FIGURE 4.4: Source weight α and Kullback-Leibler divergence estimations over different houses

improve energy disaggregation over diverse target appliances with different working patterns. For example, DW has a complex multi-stage active profile, and MV mainly operates at high

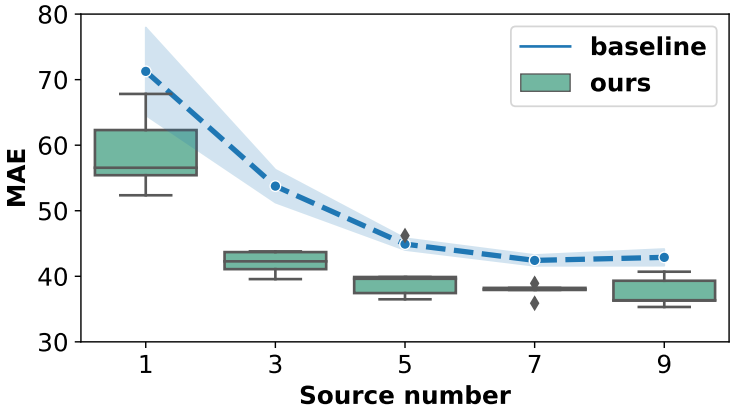


FIGURE 4.5: Estimation accuracy of HLD-MDAN for fridge with different numbers of data sources

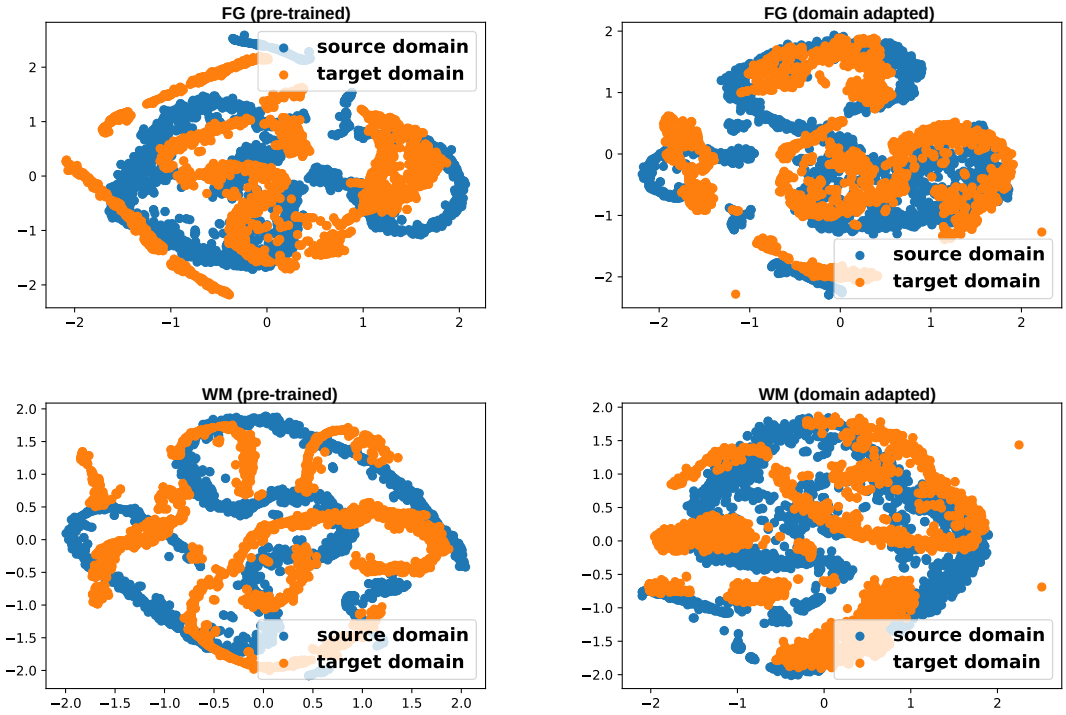


FIGURE 4.6: t-SNE visualisation of extracted feature instances before and after using HLD-MDAN for FG ($H_1^* \rightarrow R$) and WM ($H_3^* \rightarrow F$)

power consumption with a relatively simple active profile. Using HLD-MDAN, the MAE improvement can reach 26.60% and 44.99% for DW and MV, respectively. In addition, only HLD-MDAN realises lower SAE and NDE than the baseline model for MV, whereas the rest do not.

Multi-source: Our experimental datasets were built on the power readings of houses 1 and 2 from REDD, houses 1 and 2 from UK-DALE, and two randomly selected houses from REFIT. In each case, one house was marked in bold in Table 4.2 as the target, while the rest were the sources. The labelled sources that were named as H_1^*, H_2^*, and H_3^* in Table 4.3 represent the target house from REDD, UK-DALE, and REFIT, respectively. Baseline and DANN+JMMD were trained on the simple combination of all of the selected sources, while the others used multiple data sources with different weighting schemes.

In Table 4.3, we note that HLD-MDAN dominated in most cases. HLD-MDAN obtained the lowest MAE on the targets, bringing about a remarkable improvement to them. It achieved 51.53% and 58.94% MAE improvement for WM and DW with inconsistent working states. Also, for FG with low power consumption, HLD-MDAN improved MAE significantly, whilst the others performed closer to the baseline. Although the benchmarks occasionally obtained lower SAE and NDE on some targets, HLD-MDAN still offered a comparable performance. The performance consistency, however, only appears on HLD-MDAN and not on the others. We can observe the instability of the other methods due to encountering unexpected negative transfers, e.g. MDAN for DW ($H_3^* \rightarrow$ F) and AHD-MSDA for FG ($H_1^* \rightarrow$ R).

4.6.3 Analysis and Discussion

4.6.3.1 Effects of Weighting Scheme

We validated our weighting scheme in practice. In our method, we know that the distribution similarity with the target determines the α value of a given source. We estimated the Kullback-Leibler divergence (KLD) with [99] for all pairs of source and target, implicitly measuring the distribution distance. Fig. 4.4 shows the final source weights and KLD estimations. Notably, the source domain with low KLD is more likely assigned high weight. This result aligns well with the core idea of our algorithm: high weights are assigned to the sources close to the target.

4.6.3.2 Effects of Sample diversity

There is an intuition behind data analysis that we can improve the generalisation ability of a model by feeding more data samples with diverse statistical characteristics. For NILM in an unsupervised MSDA setting, we face data-inefficiency issues by naively adding more labelled houses to boost adaptation performance. We investigated the MAE of the HLD-MDAN model for FG when varying the number of data sources, shown in Fig. 4.5. We note that, whether HLD-MDAN or the baseline is applied, MAE decreases with the increase of labelled houses at the beginning, though it remains stable after the source number reaches 5. The reason is that the added houses from REFIT are far from the target when the source number is larger than 5. The data quality bounds the performance improvement on the target using HLD-MDAN.

4.6.3.3 Ablation Study

We also investigated the effectiveness of each component in our solution by comparing MDAN, AHD-MSDA and HLD-MDAN. Table 4.3 shows that MDAN performed the worst in most tasks and failed to realise domain adaptation on the target domain. According to the results of $\mathrm{H}_3^* \rightarrow \mathrm{F}$ for DW and MV, MDAN suffered negative transfer and showed severe accuracy deterioration compared with the baseline. This result is due to the inappropriate use of d_H for regression and ignoring conditional shift loss. In contrast, AHD-MSDA performs better than MDAN by replacing $d_{\mathcal{H}}$ with a tailored regression-based domain discrepancy measure similar to ours. As a case study, the result empirically verifies that $d_{\mathcal{H}}$ is not ideal for training adversarial neural networks in a regression task. Instead, RBDD is more powerful in shrinking the marginal distribution gap between domains in such an application environment. We implemented the t-SNE embeddings on the outputs from the feature extractor and compared the results of the before and after domain adaptation with RBDD, shown in Fig. 4.6. For FG and WM, distribution discrepancies between the source and target domains can be significantly reduced by applying domain adaptation. Also, the necessity of the newly introduced conditional embedding discrepancy for improving generality can be verified by comparing the performance of HLD-MDAN and AHD-MSDA. In Table 4.3, it can be seen that HLD-MDAN outperforms AHD-MSDA in most tasks. This observation indicates that

conditional loss helps the model generalise better on the target domain. Thus, we can conclude that an effective combination of all of the above losses leads to the success of HLD-MDAN.

4.7 Summary

This chapter addresses the data shortage issue for NILM model construction by leveraging multiple easily obtainable datasets with sufficient appliance-level annotations. We identify three domain shift modes on the real-world data and formulate an MSDA problem for energy disaggregation. To solve the problem, we prove a new generalisation bound of target risk given in Theorem 4.4.3 for multi-source learning. Our bound leads to a hybrid loss as 4.16 to guide model convergence by considering covariate and conditional shifts. To address hybrid loss, we propose an effective adversarial training algorithm for neural networks to optimise the learned representations, ensuring the model generalises well with the unlabelled target domain. Extensive experiments validate the effectiveness and superiority of our solution in both single-source and multi-source learning scenarios. We will deploy our design in prototype systems to test its practicality and to finetune the design.

CHAPTER 5

Federated Gradient Boosting Trees for Non-Intrusive Load Monitoring

---

Non-intrusive load monitoring (NILM) is a computational technique to allow appliance-level energy disaggregation for sustainable energy management. Most NILM models require considerable training data to capture sufficient appliance signatures for robust model fitting. However, local on-site training cannot satisfy that requirement due to limited data availability. It is thus conceivable to perform data collaboration among different stakeholders. Unfortunately, current collaborative learning approaches rely on deep learning, encryption, and differential privacy techniques associated with either expensive computation or inefficient communication. In this chapter, we propose a cost-effective collaborative learning framework, Fed-GBM (Federated Gradient Boosting Machines), consisting of two-stage voting and node-level parallelism, to address the problems in co-modelling for NILM. Through extensive experiments on real-world residential datasets, Fed-GBM shows remarkable performance in convergence, accuracy, computation and communication efficiency. The impact of hyper-parameters in Fed-GBM is also extensively studied to guide better practical use.

5.1 Introduction

This chapter consistently elucidates the constraints that arise in NILM applications within user premises, particularly in cases where local resources prove insufficient to fully realise the implementation of the solutions outlined in preceding chapters. First, as mentioned in Chapter 3, recent NILM solutions are mostly deep neural networks based [111, 5, 11]. Their complex structure and associated hyper-parameters impose a high overhead in training and inference, less suitable for devices with limited computation and storage resources. Second,

most studies [130, 92, 111] focus on long-term (more than one hour) energy disaggregation, which naturally requires a long sequence of main readings for each estimate. Managing such long readings consumes substantial storage space on local devices. Third, NILM models need considerable training data to extract representative statistical characteristics to gain high performance. However, imbalanced data often occurs at user premises due to the operating frequency of the appliances and thus leads to poor model fitting [3], especially in the cold-start phase. Although existing approaches can tackle this issue by collecting data from stakeholders for centralised model training, the potential risks of privacy leakage [120, 110] and expensive data transmission preclude them from practical use. Lastly, existing collaborative training approaches depend on encryption and differential privacy techniques to protect privacy [114, 28]. They introduce extra computation cost in model training and degrade the model performance at runtime. In addition, communication efficiency is usually neglected in recent solutions [68, 115, 119], which, however, is a noted and important factor in designing a collaborative learning system. Large communication footprints could hamper the system scalability and cause network congestion, particularly when bandwidth is limited at the user end. Thus, it is imperative to lower communication costs to support collaborative modelling activities.

In this chapter, we propose a cost-effective collaborative NILM framework, Fed-GBM (Federated Gradient Boosting Machines), to address the above challenges. By cost, we mean computational time and communication overhead consumed by collaborative learning activities. Fed-GBM integrates horizontal federated learning and Gradient Boosting Decision Tree (GBDT) training. The use of GBDT in our design is motivated by its results in non-linear regression problems with low computation complexity [23, 40]. By cooperating with the sequence-to-point (seq2point) learning, GBDT models can reach state-of-the-art accuracy for NILM without excessive computational complexity. We also perform short-term energy disaggregation by shrinking the window size used in seq2point, significantly reducing data management costs at local devices. Additionally, short-term estimation will make it possible to achieve real-time decision making on energy management. Federated learning is a technique that collaboratively trains a model across distributed devices without explicit data exchange [62]. Horizontal federated learning is a specific type of federated learning. It fits the NILM

applications where local data from different householders share the same feature space. Based on this technique, GBDT models can be jointly trained by multiple users and enhance model fitting ability locally and globally. Fed-GBM employs a two-stage voting scheme to lower communication costs by reducing the gradient information exchange during the joint model training. Our design implicitly protects users from privacy leakage while minimising the costs of using privacy protection measures. Moreover, an online scheduling strategy is proposed to enable node parallelism in model training, further improving computation efficiency and resource utilisation.

The main contributions of this chapter are:

(1) A cost-effective horizontal federated learning framework, Fed-GBM, is proposed for NILM to conduct joint GBDT model training among different electricity users. The framework addresses the data shortage and imbalance problems while matching the best-known accuracy of the centrally-trained method.

(2) We improve the model's communication efficiency by introducing a two-stage voting scheme for the horizontal federated learning environment, and the performance bound is also analysed. The benefits of privacy preservation of our model are also discussed.

(3) By leveraging characteristics of depth-wise GBDT, we explore the potential parallelism of model training and define it as an online-time Non-clairvoyant scheduling problem under precedence constraints. A node-level parallelism strategy is proposed to improve the computation efficiency of each participant under a given resource budget.

(4) We implement a prototype of Fed-GBM[1] and evaluate its performance with the residential datasets. Extensive experiments are conducted to determine the impact of hyper-parameters used in two-stage voting and node-level parallelism. The results provide a concise guide on selecting tuned hyper-parameters and let models achieve a better trade-off among different metrics in practice.

[1] https://anonymous.4open.science/r/Fed-GBM-NILM-0D27/README.md

5.2 Background

Federated learning has become a promising collaborative training approach to perform data operation among distributed data owners [126]. It can be classified into horizontal federated learning, vertical federated learning, and federated transfer learning. Most federated learning research is neural networks based [82, 13, 116], while other machine learning models are less focused. As GBDT shows remarkable success in data mining competitions, recent studies [137, 58, 68, 27, 115, 123] started integrating federated learning with GBDT. The privacy concerns of the decision tree models were first discussed in [4]. To relieve such concerns, [137] made the first attempt to enable safe tree-based distributed data mining with differential privacy. [27] and [123] proposed encryption-based strategies for GBDT in vertical federated learning. Horizontal federated GBDT learning was also presented in [68, 115]. They both performed a secure aggregation technique to alleviate privacy leakage. These solutions are costly in practice as they require extensive computation resources, and communication efficiency is not considered in the framework design.

5.3 Preliminaries

5.3.1 Gradient Boosting Decision Tree

Gradient boosting decision trees (GBDT) is a tree-based ensemble model in the family of gradient boosting machines. It comprises a set of decision trees, and each tree is responsible for attributing a given input instance x to a leaf node with a weight w, representing a prediction. The final prediction result is obtained by summing the weights of all trees. The training of GBDT is implemented in a step-wise approach. Given a set of instances, the i^{th} sample consists of a feature vector x_i and a label y_i, the goal is to minimise a regularised objective function for each tree iteratively. The objective function for the t^{th} tree is defined as

$$obj_t = \sum_{i=1}^{n} l(y_i, \hat{y_i}^{(t-1)} + f_t(x_i)) + \Omega(f_t). \tag{5.1}$$

where l is the loss function (squared error in regression), $\hat{y_i}^{(t-1)}$ is the sum of prediction results from the previous $t-1$ trees, and $f_t(x_i)$ is the weight of the leaves where x_i is assigned in the newly generated t^{th} tree. $\Omega(f_t)$ represents the regularization term used to limit the overfitting of the new tree. It can be mathematically defined as $\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$, where T is the number of leaves, w_j represents the score assigned to the leaf j, and γ and λ are hyperparameters of the penalty terms. The regularization term imposes complex constraints on the tree by limiting the number of leaves and the scores on leaves. In [26], obj_t can also be formed as a second-order approximation.

$$obj_t = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i {f_t}^2(x_i) + l(y_i, \hat{y_i}^{(t-1)})] + \Omega(f_t) \tag{5.2}$$

$$g_i = \partial_{\hat{y_i}^{(t-1)}} l(y_i, \hat{y_i}^{(t-1)}) \qquad\qquad h_i = \partial^2_{\hat{y_i}^{(t-1)}} l(y_i, \hat{y_i}^{(t-1)}) \tag{5.3}$$

where g_i and h_i represent the first-order and second-order gradients, which can be obtained by Eq.(5.3). With the gradients of all instances, the optimal weight of leaf j and objective value can be obtained by

$$w_j^* = -\frac{G_j}{H_j + \lambda},\ obj_t^* = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T \tag{5.4}$$

where $G_j = \sum_{i \in I_j} g_i$, $H_j = \sum_{i \in I_j} h_i$. I_j is a set of instances assigned to the leaf j. From obj_t^*, variance gain for each leaf can be calculated by Eq.(5.5) if a node split is performed and instances are partitioned into two subsets depicted as L and R.

$$Gain_j = \frac{1}{2}[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}] - \gamma \tag{5.5}$$

Each intermediate node is recursively split based on the selected feature and its corresponding threshold with the highest variance gain to construct a tree from root to leaf. In this process, some widely-used variants of GBDT [26, 50] employ a histogram acceleration technique to improve the efficiency of the model training.

5.3.2 Horizontal Federated Learning

Federated learning is an emerging and fast-growing collaborative training method in the artificial intelligent industry. Motivated by people's awareness of privacy protection and data control, federated learning enables data owners to perform co-modelling under the constraint that locally sensitive information is not allowed to be exposed to other participants and the central server. It can be roughly divided into three categories: horizontal federated learning, vertical federated learning, and federated transfer learning. Horizontal federated learning is widely used in cases where the datasets from different owners have the same feature space. In detail, there are n participants $\{P_1,\ldots, P_n\}$ with instance sets $\{I_1,\ldots, I_n\}$, and each instance set I_i contains full features $\{x_1,\ldots, x_s\}$. Under a given learning protocol for coordination and security, all participants collaboratively train a model without directly exposing raw data. The commonly-used horizontal federated learning framework is proposed by [81]. In this framework, model training is distributed across multiple participants and periodically aggregates the locally-trained model parameters globally. Inspired by this method, we propose a federated GBDT training framework for NILM.

5.4 Federated Gradient Boosting machines (Fed-GBM) for NILM

To improve the communication efficiency of federated learning and enable privacy protection, we design a federated learning framework named Fed-GBM for energy disaggregation among different power users. A framework overview is illustrated in Fig.5.1.

5.4.1 System Roles

Our design contains two roles: server and local runner (power user).

Server: used as a synchronisation controller to coordinate all local runners to jointly train a model of a target appliance. During the model training phase, the server receives the local

Algorithm 6 Fed-GBM Model Training

Input: Instance sets $\{I_1,\ldots, I_M\}$ contains $\{N_1,\ldots, N_M\}$ samples, located at runners $\{P_1,\ldots, P_M\}$. Each sample contains S features and a label y, λ is the stopping criterion.
Output: A trained $Model$ with T regression trees

Preprocess the samples by seq2point and Initialise their prediction values locally
for $s = 1 \rightarrow S$ **do**
    find Q quantiles for each feature
end for
for $t = 1 \rightarrow T$ **do**
    queue $\leftarrow$ [ ]: // priority queue
    queue.push(root) //root node
    for $i = 1 \rightarrow M$ **do**
        $\{(g_j, h_j)\}_{j=1}^{N_i} \leftarrow$ CalculateGrad($\{y_j\}_{j=1}^{N_i}$, $\{\hat{y_j}\}_{j=1}^{N_i}$)
    end for
    while queue is not empty **do**
        nodes= queue.pop( ) // Pop multiple nodes and run in parallel
        for node $\in$ nodes **do**
            for $i = 1 \rightarrow M$ **do**
                // G_s=$\{G_s^1,...,G_s^Q\}$, H_s=$\{H_s^1,...,H_s^Q\}$
                $\{(G_s,H_s)\}_{s=1}^{S_i} \leftarrow$ BuildHist($\{(g_j, h_j)\}_{j=1}^{N_i}$, node)
                $F_i^{TopK} \leftarrow$LocalVote($\{(G_s, H_s)\}_{s=1}^{S_i}$)
                SendtoServer(F_i^{TopK})
            end for
            //Server Executes:
            $X^{TopK} \leftarrow$ GlobalVote($\{F_i^{TopK}\}_{i=1}^{M}$)
            $\{(G_k,H_k)\}_{k\in X^{TopK}} \leftarrow$ Aggregate(X^{TopK}, $\{P_i\}_{i=1}^{M}$)
            $Split \leftarrow$ FindSplit($\{(G_k,H_k)\}_{k\in X^{TopK}}$)
            Broadcast($Split$, $\{P_i\}_{i=1}^{M}$)
            //Runners Executes:
            $children$=ApplySplit(node, $Split$, λ)
            queue.push($children$)
        end for
    end while
    Update prediction value of the samples at each P
    $Model$.append($NewTree_t$)
end for
return $Model$

training results and makes the final decision to select the model parameters that are used to split the tree nodes in the GBDT model, including features and corresponding thresholds. These global parameters will be sent to all local runners to update the model. As depicted in

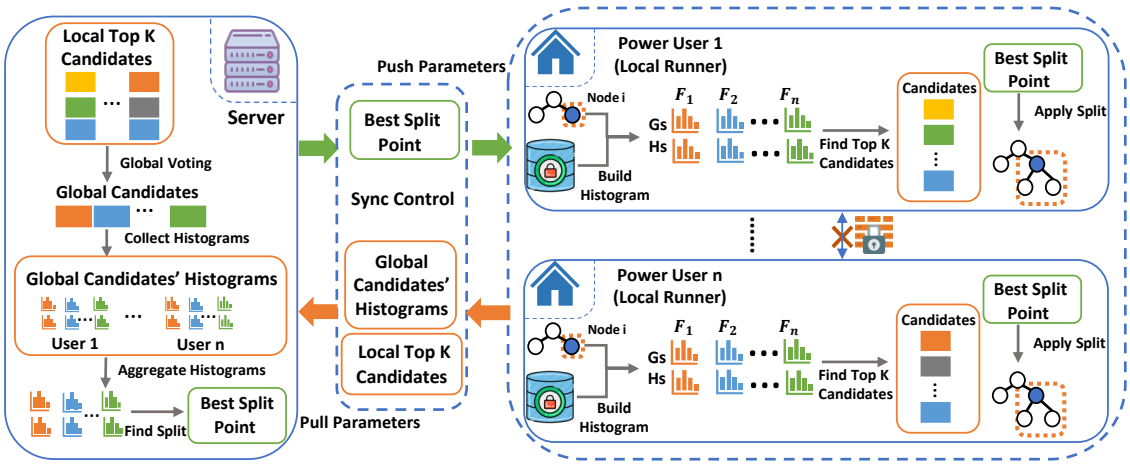


FIGURE 5.1: An overview of Fed-GBM framework

Algorithm 6, there are three core functions executed within the training process: GlobalVote, Aggregate and FindSplit.

Local runner: represents a wide range of computing devices with limited computation and storage resources located at end-user premises. The instance sets appear in different runners in a horizontal federated learning system but have the same user-defined feature space. As illustrated in Fig.5.1 and Algorithm 6, the runners participate in the entire model training. They are responsible for data pre-processing, tree structure initialisation, gradient computation, local histograms establishment, local split finding, and model updating. Local and global parameters can be exchanged via the pre-defined communication channel to perform two-stage voting in each iteration effectively.

5.4.2 The Workflow of Fed-GBM Training

The detail of the joint model training is shown in Algorithm 6, which contains six phases as shown below:

(1) **Initialise quantile sketches and prediction values**: Each local runner manages an instance set for model training, and each instance sample comprises features and a label derived from the seq2point learning paradigm. The local runners jointly create

the pre-set Q quantile sketches for different features. Then, each runner randomly initialises prediction values of local samples (line 1-4).

(2) **Initialise the new tree**: GBDT constructs an additive ensemble of trees that are trained sequentially. Any newly grown tree is constructed by recursively splitting the leaf nodes of the new tree until reaching a pre-defined stopping criterion λ. The leaf nodes to be split are stored in a priority queue. The root of the new tree is first pushed into the queue and set to active. By executing the CalculateGrad function, the runners concurrently calculate the gradients of local samples based on their corresponding labels and prediction values (line 6-10).

(3) **Build histograms**: In the process of tree growth, the active tree nodes are iteratively pulled off the priority queue. For a given tree node, each local runner P_i uses the first-order and second-order gradients $\{(g_j, h_j)\}_{j=1}^{N_i}$ to build the histograms. Each histogram corresponds to a feature s. In a histogram, data samples are partitioned into Q bins according to the created quantile sketch of s. Each bin q records the sum of gradients G_s^q and H_s^q of the samples that reside in the bin. Based on the histograms of all features, the best split point can then be found (line 16).

(4) **Find split**: We design a communication-efficient two-stage voting scheme (detailed in Section 5.4.3) to find the splits for active nodes. In this process, the runners and the server jointly search the split with maximum variance gain. Instead of searching on all features, the server collects the top-K feature candidates from each local runner and determines the global candidate features by majority voting. By aggregating the histograms of the candidate features to the global one, the server finds the best split point and then sends the result to all runners (line 17-24).

(5) **Apply split**: When the best split point is received from the server, local runners will apply the split. Two scenarios are taken into account. If a stopping criterion is met, the node stops the growth from that brunch and becomes a leaf. Otherwise, the node is split and corresponding children are pushed into the priority queue for further processing. The local runners repeat phases (3)-(5) until the queue becomes empty. When the growth of the current tree finishes, both the runners and the server will proceed to the next tree (line 26-27).

(6) **Produce the model**: When the current tree number equals the maximum number of training rounds, the system will stop training and output a NILM model. Otherwise, the training procedures of phases (2)-(5) are iteratively performed by the local runners and the server until reaching the threshold.

5.4.3 Two-stage Voting for Fed-GBM

Some existing works [115] follow the idea of traditional data parallelism algorithms to enable horizontal federated learning for GBDT; the histograms of all features are built at local runners and then sent to the server for further processing. Assuming no message exchange happens between runners and no node parallelism is involved, the total communication cost is $(MSH2^D)$, where M is the number of runners, S is the number of features, equal to the size of the selected window in the seq2point training paradigm, H is the size of a histogram, and 2^D represents the node number of a binary tree where D is the depth of a depth-wise tree. The cost is proportional to S when other factors are fixed. As network traffic is highly dynamic, the large message overhead could weaken such systems' robustness and cost efficiency.

To reduce communication costs while preventing information leakage, we propose a two-stage voting scheme for horizontal federated learning and prove its feasibility in Section 5.4.4. In our approach, instead of performing the split finding over the whole feature space, only the candidates who pass the local voting stage are entitled to engage in the global search at the server. The communication overhead is thus significantly reduced, as well as the risk of information leakage, discussed later in Section 5.4.6.

Voting at local runner: For the locally-built histograms, all runners execute the function LocalVote in Algorithm 6 to perform split finding in parallel. In the searching process, all features and corresponding split thresholds are iteratively evaluated by calculating variance gains. According to the maximum variance gain, the runner selects top-K candidate features and sends the result to the server. The result is a list of feature indexes. So the communication cost in this step is (MK).

Voting at server: In the split finding phase, the server first stays in a synchronous barrier and waits to collect the local voting results from all runners. Once all results are fully retrieved, the server performs global voting among all received candidates. In this step, the server selects a list of βK (β>1 & $\beta \in \mathbb{N}^+$) candidate features by using majority voting. The features are ranked based on the times chosen by runners, and the candidates with a higher majority win. For a given feature, there is a sufficient condition to ensure it is selected into the global feature list by majority voting, which is provided as a lemma and will be used in Section 5.4.4.

LEMMA 8. *If a feature f is selected by no less than $[\frac{M}{\beta}+1]$ local runners in the local voting, it must be contained in the global top-βK feature candidates.*

PROOF. We prove this lemma using proof by contradiction. Assuming no less than $[\frac{M}{\beta}+1]$ runners select a feature f, but this feature is still not one of the global top-βK candidate features. So, minimum βK features are selected by at least $[\frac{M}{\beta}+1]$ runners. The top-βK features use no less than $(\frac{M}{\beta}+1)\beta K$=$KM$+$\beta K$ votes. As all local runners have only KM votes, there is a contradiction in this case. Thus, f is one of the global top-βK feature candidates. □

Global Split Finding: After globally selecting candidate features, the server first waits to collects the histograms of these features from all local runners and aggregates the histograms of the same feature to a global one. Then the function FindSplit in Algorithm 6 is executed to search for the best split feature and corresponding split point.

In this manner, only the histograms of K features need to be sent by runners, and the total communication cost of our design is $(MKH2^D + MK)$, which is independent of S. In practical use, a proper value of $\frac{K}{S}$ is the key to achieving the desired tradeoff between communication efficiency and model accuracy.

5.4.4 Theoretical Analysis for Two-stage Voting

This part gives a theoretical analysis to ensure a probability bound of finding the optimal global split using the two-stage voting scheme, which reflects the performance gap with the

traditional centralised searching scheme. With the lower bound, we show that the two-stage voting scheme is of practical importance, and the gap can be effectively reduced by more users engagement in co-modelling. Now we provide the probability bound by a theorem as follows:

THEOREM 5.4.1. *Suppose that m local runners $\mathbb{P}$=$\{P_1, ..., P_m\}$ manage different sizes of datasets $\{N_1, ..., N_m\}$ in a horizontal federated learning system, and follow the two-sage voting scheme to train a GBDT model. Let local voting size to be K and global voting size to be βK (β>1 & $\beta \in \mathbb{N}^+$). For an arbitrary tree node, the feature with the largest variance gain can be found in full feature space S with a probability of at least*

$$\sum_{n=[\frac{m}{\beta}+1]}^{m} \sum_{t=1}^{C_m^n} \prod_{P_i \in R_t} \left(1 - \left(\sum_{j=K+1}^{S} \delta_{(j)}(N_i, K)\right)\right) \prod_{P_i \in \mathbb{P}/R_t} \left(\sum_{j=K+1}^{S} \delta_{(j)}(N_i, K)\right) \tag{5.6}$$

In (5.6), $\delta_{(j)}(N_i, K) = \alpha_{(j)}(N_i) + 4e^{-c_{(j)}N_i\left(l_{(j)}(K)\right)^2}$, where $l_{(j)}(K)$ indicates the distance between the largest variance gain and the j^{th} largest one for any $j \geq K+1$, Also, $\lim_{N_i \to \infty} \alpha_{(j)}(N_i) = 0$, and $c(j)$ is constant. $R_t \subseteq \mathbb{P}$ denotes a set of the runners that select the best split feature in the local top-K list.

PROOF. To better understand the proof, we first introduce some necessary notations. For a given tree node, the top-K ranked list of features by the variance gain is L_i at any runner P_i, and the global feature list over all runners by majority voting is L'. The feature with the largest variance gain is f^{max}. We denote n as the number of runners selecting f^{max} as a candidate feature. In this chapter, there are two steps to prove the theorem. First, we calculate the lower bound of probability for the case any given runner P_i selects f^{max}, denoted as $P(f^{max} \in L_i)$. We know that households from the same geographic region share similar appliance signatures and usage patterns [10, 34]. As our approach serves the same application scenario, we let datasets in different runners share the same data distribution. Under such conditions, we follow the mathematical proof of Theorem 4.1 in [85] and easily derive the lower bound of $P(f^{max} \in L_i)$ as following:

$$P(f^{max} \in L_i) \geq 1 - \sum_{j=K+1}^{S} \delta_{(j)}(N_i, K) \tag{5.7}$$

Second, based on the result from the first step, we can calculate the lower bound of probability for the case where best split feature is selected into the global candidate list at the server, represented as $P(f^{max} \in L')$. According to Lemma 8, $n \geq [\frac{m}{\beta} + 1]$ is a sufficient condition for $f^{max} \in L'$ to be satisfied. $P(n \geq [\frac{m}{\beta} + 1])$ can be calculated as follows:

$$\begin{aligned} P(n \geq [\frac{m}{\beta} + 1]) &= \sum_{n=[\frac{m}{\beta}+1]}^{m} P(n) \\ &= \sum_{n=[\frac{m}{\beta}+1]}^{m} \sum_{t=1}^{C_m^n} P(R_t | R_t \subseteq \mathbb{P}, |R_t| = n) \\ &= \sum_{n=[\frac{m}{\beta}+1]}^{m} \sum_{t=1}^{C_m^n} \prod_{P_i \in R_t} P(f^{max} \in L_i) \prod_{P_i \in \mathbb{P}/R_t} P(f^{max} \notin L_i) \end{aligned} \tag{5.8}$$

By combining Ineq.(5.7) and Eq.(5.8), we can get the lower bound of $P(f^{max} \in L')$ in Theorem 1 . □

REMARK. By Theorem 1, our proposed training scheme can guarantee that, for a given node, the best split feature can be selected with a probability of at least (5.6). From a statistical viewpoint, we can improve the accuracy of the GBDT model by boosting the probabilistic lower bound of finding the optimal split for each node during the tree growth. Meanwhile, the theorem presents users with a clear path to improving the model's accuracy by adjusting the parameters accordingly. The size of local training data N_i, voting size K, and the number of local runners m are the three key parameters that could affect the model's accuracy. By combining (5.6) and the definition of $\delta_{(j)}(N_i, K)$, we can observe that fixing the other parameters to increase N_i, the lower bound of $P(f^{max} \in L')$ also increases since $\delta_{(j)}(N_i, K)$ decreases. Especially when the data is sufficient in the local runners, the local voting result is similar to the global voting one and $P(f^{max} \in L_i) \rightarrow 1$. Similarly, the lower bound also increases with the increase of K. But if N_i is large enough, K becomes insensitive to the lower bound change, and thus we can select a small K to reduce the communication cost

during the model training. The increase of m also contributes to improving the lower bound. With more participators, the empirical distribution of training data will closely approximate the population distribution, and the lower bound will approach 1. However, it could result in higher synchronisation overhead and aggravate the risk of server crashes. We can leverage the above findings in real applications to get a fine-tuned parameter combination to meet diversified customers' demands.

5.4.5 Node Parallelism for Tree Growth

As the workload of splitting a node is small, the available CPU cores cannot be fully utilised at both local runners and the server. It results in the waste of computation resources. To address such an issue, we propose a node-level parallelism scheme for tree growth.

During the training, any two nodes in the queue have no parent-child relationship, since only when an active node is split and deactivated their children can be added to the queue. So there is a chance to achieve parallelism among different nodes during the model training. In our system design, there are w executors with identical resources running in parallel at runners and the server. The value of w is limited by the maximum computation capacity of local runners. The jobs of node split, issued during the tree growth, are assigned among these executors for processing. Before introducing a feasible solution to job scheduling, we first identify the scheduling problem in the node-level parallelism scheme by the following proposition.

PROPOSITION 1. Given w executors with identical resources for jointly training a GBDT model on the Fed-GBM framework, the job allocation under node-level parallelism setting is a Pw | online-time-nclv, prec, r_j | C_{max} scheduling problem.

PROOF. In the node-level parallelism scheme, w executors with identical resources run parallel at a local runner or the server. They can be seen as w parallel machines, represented by Pw. The scheduler needs to determine when and where the continuously released jobs are executed during tree construction. The jobs have precedence constraints, represented by $prec$. Apart from the root node, any tree node is generated by its parent node. This makes its

corresponding split job j released only after its parent job i is finished. The release time of the job j denoted as r_j is equivalent to the completion time of the job i. For the online-time-nclv scheduling problem, where nclv is short for Non-clairvoyance, the scheduler is not aware of the existence of a job until its release and also has no information about the processing time of that job at its release date. Since the jobs are dynamically generated during the tree growth, their processing time is unknown when released and is hard to be estimated. It is thus to be formed as an online-time-nclv problem. Besides, our objective is to minimise the makespan C_{max} (total time cost) of the tree construction. Overall, the job allocation can be described by using the three-field Graham's notation [42] as a *Pw* | online-time-nclv, prec, r_j | C_{max} scheduling problem when the node-level parallelism is enabled for Fed-GBM. □

The List Scheduling (LS) algorithm proposed by [41] achieves the best-provable competitive ratio of the online-time-nclv scheduling problems. LS assigns a released job for node split to the executor that has the least load among the executors. Compared with optimal offline algorithm (OPT), the following ratio always holds

$$\frac{C_{max}(LS)}{C_{max}(OPT)} \leq 2 - \frac{1}{w},\ w \geq 2 \tag{5.9}$$

At both local runners and the server, LS assists in optimising the fair sharing of executors over split jobs with non-trivial dependencies and reaches the best performance ratio of (2-$\frac{1}{w}$). Our node parallelism design significantly improves computation efficiency and resource utilisation for model training with this approach.

5.4.6 Discussion on Privacy Leakage

In our training system, no peer-to-peer communication is allowed. This design means attackers can not pretend to be participants to directly retrieve sensitive data from other users in the learning process. A communication channel is established only between each runner and the server. There are three places where the privacy leakage potentially occurred. First, local voting results need to be uploaded to the server for global voting. Except for the partial feature indexes, no other informative knowledge is exposed. Second, the server collects histograms of

global feature candidates to perform split finding. For any immediate node, only partial local samples are involved in the histogram construction. Using two-stage voting, the number of histograms is a constant K, and locally-selected candidate features are dynamically changed at different nodes. So it is hard to obtain histograms of full features with all data samples, and the risk of privacy leakage is significantly relieved, especially when the K is far smaller than the total feature number. Third, the global result is learned by all the runners. The result only contains the split information for a certain node, the best split feature, and the corresponding threshold. Based on such information, the runners know the direction to split the node. Malicious runners cannot deduce any sensitive information of any users from it.

5.5 Experiments

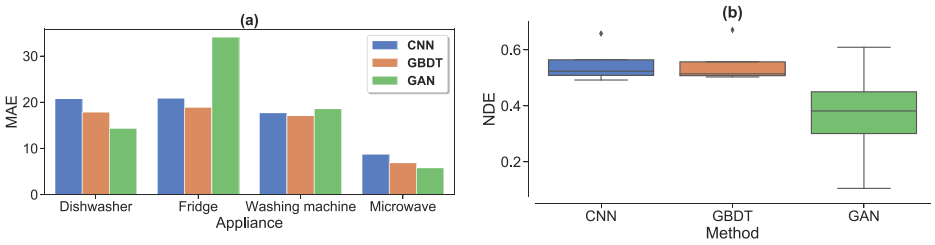


FIGURE 5.2: Comparison of performance over different training methods

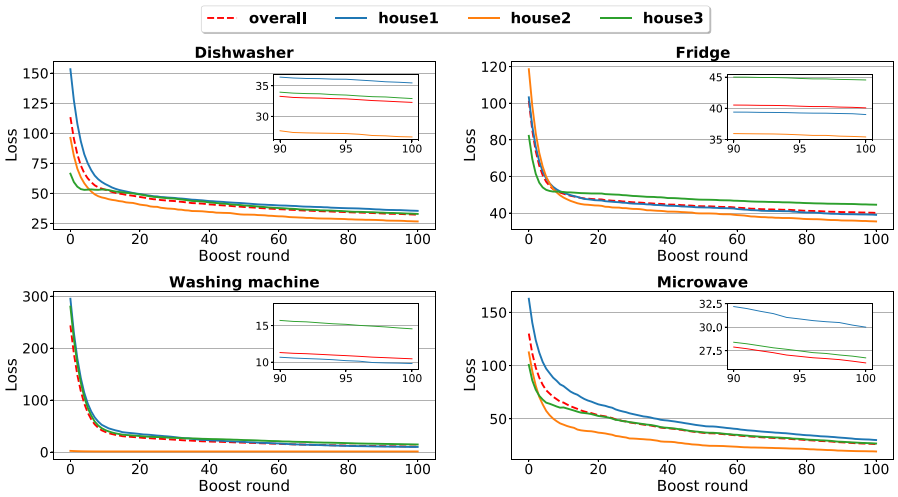


FIGURE 5.3: Convergence of training loss for Fed-GBM on REDD

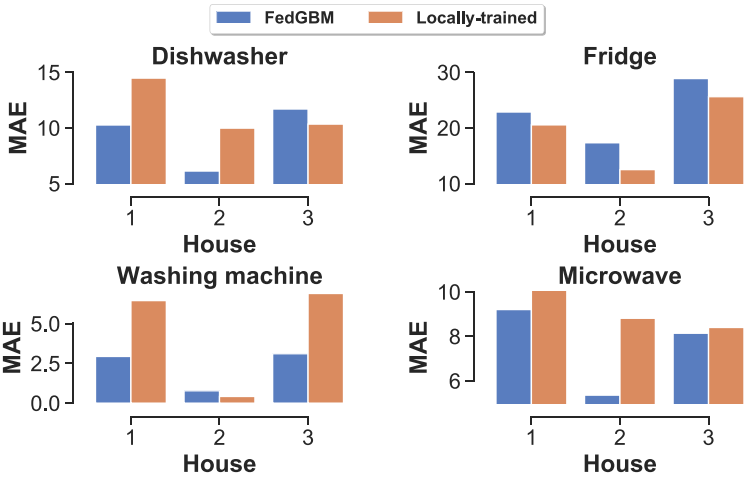


FIGURE 5.4: Comparison of MAE across three houses between locally-trained methods and Fed-GBM on REDD

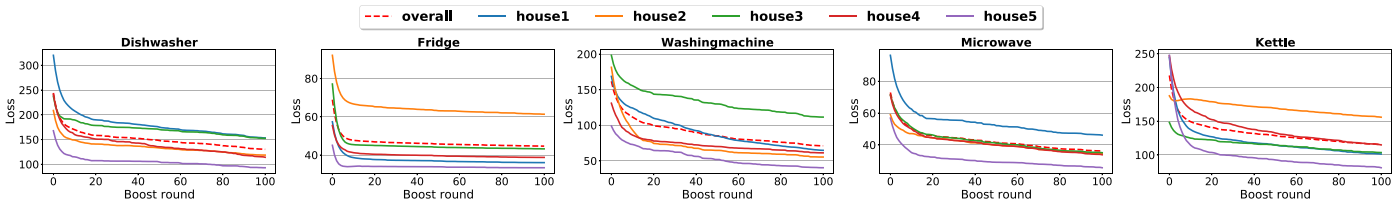


FIGURE 5.5: Convergence of training loss for Fed-GBM across five houses on REFIT

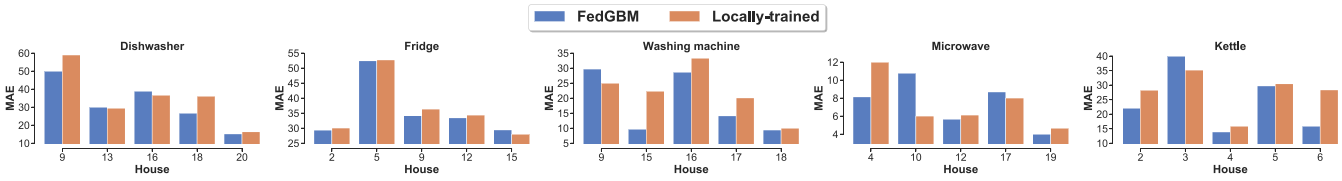


FIGURE 5.6: Comparison of MAE across five houses between locally-trained methods and Fed-GBM on REFIT

5.5.1 Experimental Setup

(1) **Datasets**: In the experiments, we employed three open datasets, UK-DALE, REDD, and REFIT, each containing the residential low-frequency appliance-level and aggregated power consumption. Our experimental datasets are built from the power readings of houses 1, 2, and 3 from REDD, building 1, 2 from UK-DALE, and five randomly selected houses from REFIT. For all datasets, we sampled the active load every 8 seconds. Commonly-used appliances were chosen to implement model

training, such as dishwasher, fridge, washing machine, and microwave. For each dataset, 80% of samples were used for model training, and the remainder for testing.

(2) **GBDT Implementation**: To better evaluate the performance of GBDT for NILM, we constructed different models for four common appliances from UKDALE and two state-of-the-art neural networks, CNN and GAN, as performance indexes. All models were constructed and tested based on the same dataset to allow fair comparison. We implemented the architecture of CNN and GAN based on [34] and [5] respectively. The hyper-parameters for training different NILM models are provided in Table 5.1. In these experiments, the window sizes of seq2point were set to 19 to study the short-period energy disaggregation on the selected appliances. These three algorithms were implemented on a desktop computer with Intel i7 8700 CPU @2.81GHz, 32GB DDR4 RAM, and NVIDIA GTX 1080 Graphics Card with 8GB GDDR5X VRAM.

TABLE 5.1: The parameters for training NILM models

Hyper-parameters for training CNN

Maximum epochs	100	Learning rate	0.001
Batch size	1024	Beta1	0.9
Early-stopping epochs	10	Beta2	0.999
Optimizer type	Adam	Epsilon	10^{-8}

Hyper-parameters for training GAN

Maximum epochs	100	LR for generator	0.00001
Batch size	1024	LR for discriminator	0.00001
Early-stopping epochs	10	Beta1	0.5
Optimizer type	Adam	Beta2	0.999

Hyper-parameters for training GBDT

Maximum boosting round	100	Learning rate	0.232
Maximum depth	10	L1 regularisation	0.0214
Maximum bins	500	L2 regularisation	0.0001

(3) **Fed-GBM Implementation**: Fed-GBM was implemented in Python and deployed on a scalable distributed Dask Fargate cluster connected to a powerful EC2 server from AWS. Dask is an open-source lightweight distributed computing library, which enables the control of data locality and management of task scheduling [107]. The workers represent the distributed power users, each equipped with 4 vCPU and 4GB RAM. The EC2 server coordinates different workers to train a unified GBDT model

based on our proposed algorithm. The specification of the server is 72 vCPU, 144GB RAM, 2 x 900 NVMe SSD, and 25Gbps network bandwidth.

(4) **Evaluation Metrics**: We also used three commonly-used metrics for performance evaluation of NILM models, including MAE, SAE, and NDE. For all three metrics, the lower the value, the more minor deviation between estimates and ground truth generated by the model. We also defined a scoring metric to assess different values of hyper-parameter K used by two-stage voting. It reflects the trade-off among accuracy, training overhead, and information leakage in Fed-GBM. The scoring metric and relevant notations are defined as follows:

$$score^K = \frac{Gain^K}{Loss^K} = \frac{\gamma \cdot TOG^K}{\alpha \cdot AL^K + \beta \cdot PL^K},\ \alpha + \beta = 1,\ \alpha, \beta \in (0, 1) \tag{5.10}$$

$$AL^K = MAE^K - MAE^{K^{max}},\quad TOG^K = TC^{K^{max}} - TC^K \tag{5.11}$$

where TC^K denotes training cost, AL^K , PL^K and TOG^K are accuracy loss, privacy loss and the gain of training overhead when K is assigned for two-stage voting. Since PL^K=$\delta(K)$ is proportional to the value of K, we directly use K to quantify privacy leakage in our experiment. α, β, γ are the weights assigned to AL^K , PL^K and TOG^K respectively, showing the importance of each factor. They need to be pre-defined based on the condition depicted in Eq.(5.10). The higher score a model obtains, the better trade-off it can achieve in practical applications.

5.5.2 Performance of GBDT for NILM

TABLE 5.2: Performance of Different Models for Energy Disaggregation

Appliance	CNN (seq2point)			GBDT			GAN		
	MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE
Washing machine	17.71	0.011	0.51	17.11	0.0034	0.52	18.59	0.635	0.37
Fridge	20.87	0.022	0.53	18.86	0.0003	0.51	34.11	0.334	0.61
Dishwasher	20.79	0.176	0.49	17.86	0.0042	0.50	14.34	0.105	0.11
Microwave	8.74	0.127	0.66	6.89	0.0025	0.67	5.79	0.158	0.40
Mean	17.03	0.084	0.54	15.18	0.0026	0.55	18.21	0.308	0.37

TABLE 5.3: Comparison of average performance on model size, training and inference costs over three models on UK-DALE

Method	Model Size(KB)	Inference Time(s)	Training Time(s)
CNN	11935	82.50	5621.41
GBDT	830	8.09	162.42
GAN	182896	212.40	11722.01

To validate the effectiveness of GBDT on NILM, we comprehensively examined the accuracy and running costs by different metrics and compared it with the CNN-based and GAN-based solutions. The performance comparison results are shown in Fig. 5.2 and Table 5.2. According to Fig. 5.2(a) and Table 5.2, we observe that GBDT outperformed CNN on all target appliances when the three performance metrics are jointly considered. GAN provided the best accuracy on the dishwasher and microwave but had the worst performance on the others. This result suggests that the performance of the GAN model varies over appliances. It can also be validated from Fig. 5.2(b), which shows the NDE distribution of three methods on different appliances. NDE values of both GBDT and CNN vary in a limited range (0.4 to 0.6), while large variance occurs in the GAN model (0.1 to 0.6). We can thus conclude that GBDT and CNN provide stable effectiveness for different target appliances, but GAN shows low stability and is inappropriate for residential applications. In Table 5.2, we also noticed that the average SAE of GBDT is only 0.0026, far better than the other two models. It indicates that GBDT can produce the near-ground-truth energy estimation. Additionally, we investigated the average model size, training time and inference time of three different models on the selected appliances. As shown in Table 5.3, both CNN and GAN models incur large storage and computation costs. The average model sizes of two neural networks reach 11935KB and 182896KB, 14$\times$ and 220$\times$ larger than GBDT, respectively. Under the same hardware settings, the average training time of GBDT is only 162.42s, 34$\times$ faster than CNN. The GAN shows the largest training cost, which is 11722s since both generator and discriminator have to be trained for energy disaggregation. Given the same data size for testing, GBDT is 10$\times$ and 26$\times$ faster than CNN and GAN in inference time. Overall, GBDT significantly outperforms the other two neural networks for NILM applications regarding performance and resources needed, making it more applicable for a wider range of devices.

5.5.3 Overall Performance of Fed-GBM

TABLE 5.4: Comparison of estimation accuracy of different methods on REDD

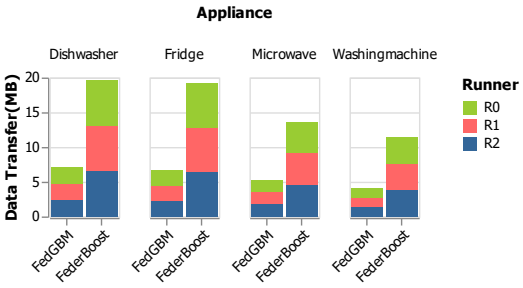
Appliance	Locally-trained			Fed-GBM			Centrally-trained		
	MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE
Dishwasher	11.78	0.020	0.55	9.60	0.016	0.44	9.58	0.006	0.43
Fridge	23.39	0.0006	0.45	22.17	0.0002	0.44	20.04	0.0003	0.44
Washing machine	4.90	0.013	0.27	2.66	0.005	0.16	2.39	0.004	0.15
Microwave	10.85	0.019	0.62	9.02	0.014	0.46	7.75	0.008	0.41

TABLE 5.5: Comparison of estimation accuracy of different methods on REFIT

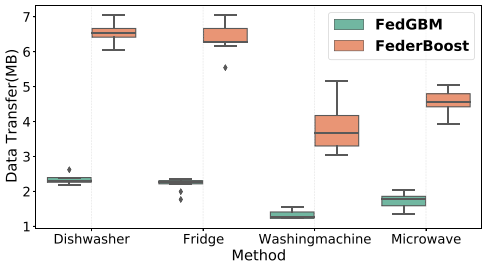
Appliance	Locally-trained			Fed-GBM			Centrally-trained		
	MAE	SAE	NDE	MAE	SAE	NDE	MAE	SAE	NDE
Dishwasher	36.39	0.020	0.79	32.84	0.009	0.762	32.74	0.012	0.77
Fridge	37.03	0.004	0.72	36.53	0.007	0.733	36.50	0.004	0.73
Washing machine	22.44	0.104	0.85	19.67	0.064	0.782	18.66	0.069	0.74
Microwave	7.46	0.098	0.91	7.51	0.073	0.865	7.35	0.049	0.88
Kettle	27.67	0.048	0.79	24.63	0.016	0.771	23.92	0.042	0.76

In this experiment, we evaluated the effectiveness and the generalisation ability of Fed-GBM on REDD and REFIT. As mentioned, we selected three houses from REDD and five random houses from REFIT. Both convergence and accuracy were evaluated.

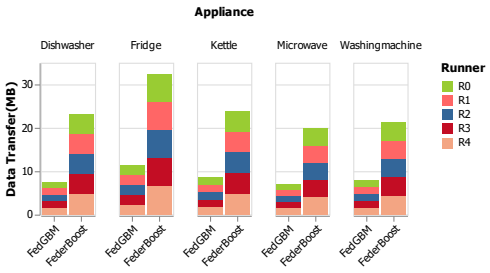
Convergence: The loss convergences of the Fed-GBM models for two residential scenarios are shown in Figs. 5.3 and 5.5. For each target appliance model, we studied the overall training loss and the loss in each house. As shown in Fig. 5.3, except for the performance of the washing machine model on house 2, the training loss across other target appliances drops with the increase of boost rounds. As the washing machine is infrequently used in house 2 and the portion of data reflecting active operation status is extremely low, the downward trend of training loss is not obvious compared with other houses. Although the initial and final losses of the three houses are different due to the divergence of usage patterns among different users, it is easy to observe that the convergence rates of the three houses are similar to the overall one. In REFIT, more power users contribute to the co-modelling, and more target appliances engage in energy disaggregation. As depicted in Fig. 5.5, Fed-GBM can still provide stable overall loss convergence across five target appliances without sacrificing anyone's performance. The result illustrates that both overall and local convergences are



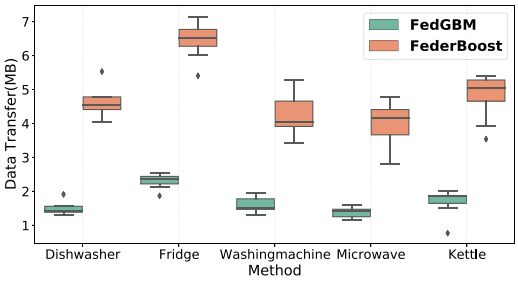
(a) Combined outgoing traffic on REDD



(b) Box plot on REDD



(c) Combined outgoing traffic on REFIT



(d) Box plot on REFIT

FIGURE 5.7: Comparison of outgoing traffic by using Fed-GBM and FederBoost for different appliances. (a) and (c) show the combined communication cost of different local runners for building a tree on REDD and REFIT. (b) and (d) are the corresponding box plots, showing the outgoing traffic distribution over different trees.

guaranteed by Fed-GBM regardless of appliance type, appliance number, and residential environment.

Accuracy: Our implemented Fed-GBM models were compared with the locally-trained ones and centrally-trained ones on different appliances. The comparison results over three main performance metrics are provided in Tables 5.4 and 5.5. For both REDD and REFIT, the centrally-trained models often achieve the best performance, while the locally-trained ones always show the worst. This result is because the centralised training method can easily extract sufficient representative knowledge and identify statistical patterns from all power users. It also ensures the natural generalisation of the models across different houses in practical use. In contrast, locally-trained models, built on limited training data, cannot sufficiently capture common appliance signatures, especially when the usage frequency of the target appliance is low. Similar to centrally-trained models, Fed-GBM models outperform locally-trained ones. They provide a close approximation of MAE to the centrally-trained method and

also prevent information leakage from individual power users. Moreover, some Fed-GBM models achieve lower SAE or NDE values than the centrally-trained ones, such as in the case of the dishwasher and kettle in REFIT. It means Fed-GBM sometimes provides a better estimation of total energy consumption and helps avoid outlier estimates. Apart from the overall performance evaluation, accuracy examination is performed on individual household owners from REDD and REFIT. The result is presented in Fig. 5.4 and 5.6. We observed that the MAE value of Fed-GBM is lower than that of local ones for most households. Fed-GBM shows remarkable adaptability across different power users. In brief, the experimental results verify that Fed-GBM has superior performance against the other two training approaches when both privacy protection and accuracy are jointly considered.

Communication Efficiency: The improvement of communication efficiency is also carefully examined on Fed-GBM compared with a state-of-the-art horizontal federated GBDT algorithm, FederBoost [115]. As communication efficiency is the main focus, we implemented the FederBoost prototype without involving its secure aggregation and differential privacy components in our tests. Both algorithms were examined under the same network settings to make a fair comparison. For the tree construction, we monitored the outgoing traffic of each local runner on REDD and REFIT. The results are provided in Fig.5.7. As depicted in Fig.5.7 (a) and (c), the combined outgoing traffic of FederBoost is more than two times of Fed-GBM. The communication cost of each runner shows the same result. According to Fig.5.7 (b) and (d), the maximum outgoing traffic of Fed-GBM is less than 3MB for any appliance, while FederBoost averagely generates more than 4MB traffic during the tree growth. In addition, Fed-GBM remains at a stable communication cost since the outgoing traffic varies within a narrow range. However, FederBoost shows a significant variance in communication cost, so the unbalanced communication cost could often occur over different trees. Overall, Fed-GBM reduces communication overhead and balances the outgoing traffic over different trees.

5.5.4 Performance of Node-level Parallelism

We evaluated the node-level parallelism of Fed-GBM by implementing model training with different executors on REFIT. The results are shown in Fig.5.8. Compared with the sequential

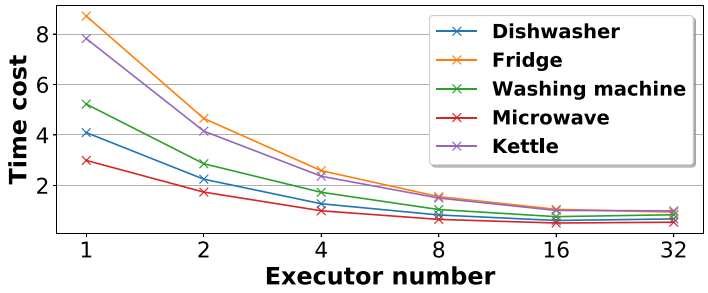


FIGURE 5.8: Comparison of training overhead over different number of executors for five appliances

training process on a single executor, there is a significant reduction of training cost along with the increment of the executors before reaching 8. When the number is greater than 8, the time cost reduction will gradually decrease. When reaching 16, the increment of executors commences incurring more time cost. The experiment result suggests our design can fully utilise the computation capacity of devices. However, the large number of executors can cause resource underutilisation at the initial layers of the tree as not that many jobs require service. This negative effect on training efficiency will become obvious if the executor number exceeds a threshold. Therefore, we have to carefully consider the dual effect of the executor number and find an optimal value for the node-level parallelism in practical use.

5.5.5 Analysis of the impact of K value

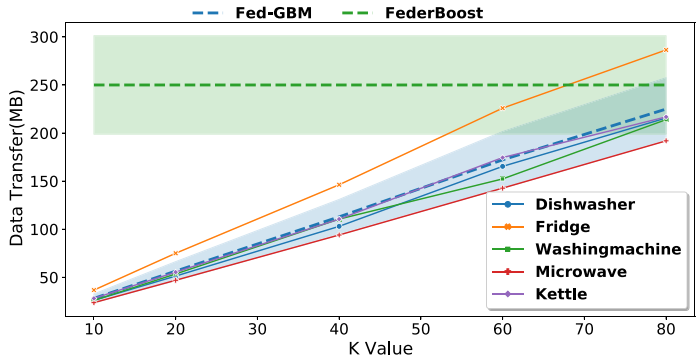


FIGURE 5.9: Impacts of K on communication efficiency for Fed-GBM. The blue dash line represents the outgoing traffic of Fed-GBM with a 95% confidence interval, while the green dash line shows the result of FederBoost.

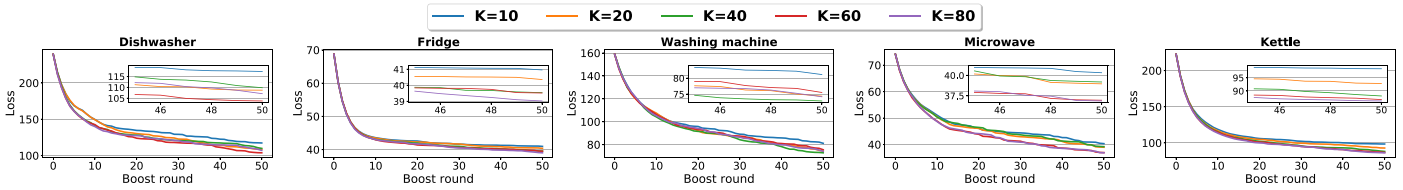


FIGURE 5.10: Impacts of K on loss convergence for Fed-GBM

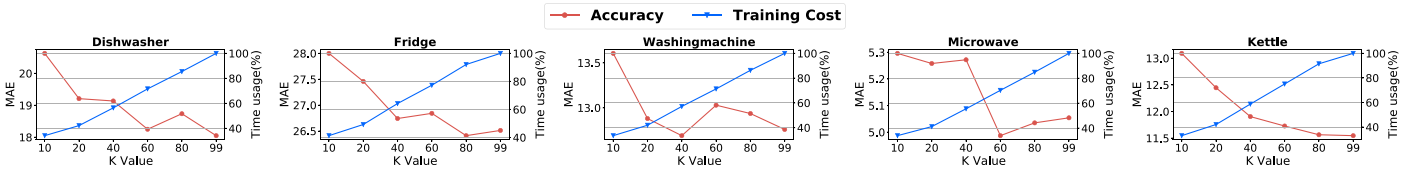


FIGURE 5.11: Impacts of K on accuracy and training cost for Fed-GBM

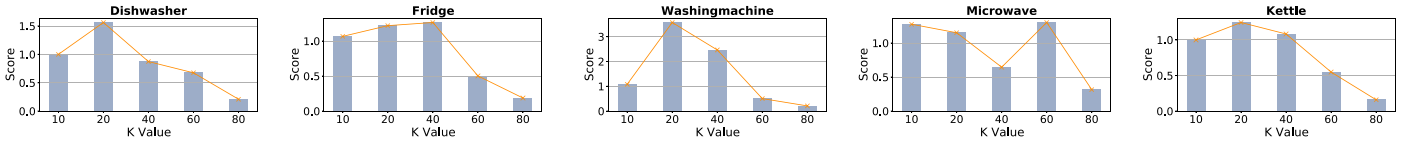


FIGURE 5.12: The score comparison of different K values

To better understand the impact of the K value for Fed-GBM, we carried out model training with different K values on five selected appliances from REFIT. The window size for seq2point is extended to 99 to demonstrate the scalability of our approach. It is also the acceptable maximum K value for short-term NILM. We first examined the impacts of K on communication efficiency for Fed-GBM. As shown in Fig. 5.9, for any appliance model, the total outgoing traffic of local runners increases with the K value. The blue dash line depicts the arithmetic mean of transferred data volumes overs five appliance by using Fed-GBM while the green dash line represents the result of FederBoost. It is easy to note that the average communication cost of Fed-GBM is proportional to K but does not exceed FederBoost, which complies with the theoretical analysis in Section 5.4.3. Fig. 5.10 shows the convergence of training loss for different K. For any target appliance, the downward trends of the models with different K are similar when the boost rounds increase. Thus we can tell that the convergence of Fed-GBM is not sensitive to the K value. Finally, accuracy and training cost are another two critical factors that should be assessed when determining K value for model construction. The details of evaluation results are given in Fig. 5.11 which

shows the time cost of model training rises with the increase of the K value. This contributes to the incremental communication cost and workloads for split finding. However, for most appliances, the variance of MAE is not a definite downward trend as the K value increases. There is no linear correlation between MAE and K. The lowest MAE of dishwasher and microwave occurs when K is 60, while both fridge and washing machine show the highest estimation accuracy when K is 40. For the kettle, MAE decreases with the increase of K. Thus, it is interesting to note that MAE does not always decrease when K increases. A low K value in the setting of two-stage voting could bring more accurate estimates.

Based on the metric defined in Eq.(5.10), we can also evaluate the performance of Fed-GBM by jointly considering training cost, accuracy, and privacy protection when different values are assigned to K. To be specific, Fig.5.12 demonstrates the impact of K for the final score assessment. The K value associated with the highest score differs in five appliances, but the peak often occurs at 20. The score will drop for most appliances when K is greater than 40. Thus, there is a high probability that $\frac{K}{S}$, valued between 20% and 40%, achieves the highest score and the best trade-off between the factors for the NILM model training.

5.6 Summary

This chapter proposes a collaborative learning framework, Fed-GBM, to address the low model fitting of local on-site training due to the limited data scale in NILM applications. In Fed-GBM, a two-stage voting scheme is proposed to simultaneously realise communication cost reduction and privacy protection. Node parallelism for tree growth is also implemented based on an online scheduling algorithm to improve resource utilisation. Our technical designs significantly reduce the training costs for adapting different levels of resource provision. We also conducted comprehensive experimental studies for Fed-GBM, and it shows remarkable performance in all tests. The impact of the K value for two-stage voting and executor number for node-level parallelism is also extensively studied to guide the hyper-parameter selection in practical use.

CHAPTER 6

Conclusion

This thesis discusses the benefits of using intelligent data analysis to improve energy management and describes two pressing challenges emerging in ML-based predictive analysis applications. The first of these issues is the lack of a cost-effective approach suitable for devices with limited resources. The second is the data scarcity issue, which leads to the failure of model fitting on target prediction tasks.

Energy management is an online scheduling problem requiring prompt energy-related knowledge estimation. As such, predictive analysis in the energy field can be categorised as a time-sensitive task and needs on-premises implementation. Besides, advanced deep mining techniques increase the potential risk of privacy leakage. This means that users typically prefer to place the training and inference task in the local devices in case information breaches occur during run time. Existing powerful ML-based models, however, contain large-scale hyper-parameters and impose high computation and storage costs at run time. Resource-constrained local devices poorly accommodate such models with complex structures. Toward this end, Chapter 2 and 3 propose lightweight learning solutions for two predictive analysis applications, photovoltaic power prediction and non-intrusive load monitoring. We leverage the advantages of state-of-the-art implementation with gradient boosting decision trees (GBDT) to lower computational costs and improve prediction accuracy. Since time series are involved in both applications, we introduce two similar data preprocessing strategies to fully capture the latent temporal patterns from a long-term time series. Such data augmentation approaches further enhance performance guarantee. Chapter 5 creates a federated version of GBDT for joint NILM modelling. This work takes a step toward lowering communication costs and

improving resource utilisation, extending the use of GBDT to wider hardware settings and various environmental constraints.

In this research, considerable research efforts have been dedicated to addressing the data shortage issue for energy management. Although labelled data from the real scene is precious and necessary for model training, it is also labour-expensive to acquire. As such, transfer learning and federated learning are promising techniques for solving target learning tasks by leveraging more easily obtainable datasets from other sources. Chapter 3 presents a unique source-free knowledge transfer approach using gradient boosting trees to alleviate the impact of data drift, whilst also protecting source data from privacy leakage. We also use feature importance derived from Shapley Additive Explanations (SHAP) to quantify the transferability of each feature. Based on the transferability, only partial nodes need optimal searching, and the whole model structure remains, which significantly reduces computation costs. Chapter 4 theoretically explores the driven factors that affect transferability between source and target tasks and presents a multiple source domain adaptation (MSDA) algorithm based on our newly proven generalisation bound of target risk. Our defined domain discrepancy fully considers both marginal and conditional shifts, which are potentially detrimental to model performance on target tasks. To reduce such domain discrepancy, an adversarial learning strategy is applied to optimise model parameters for learning task-discriminative and domain-invariant features. Our solution extracts the statistical relationships between labelled source data and unlabelled target data. Thus, our target models can address detrimental sources by adaptively assigning high weights to the source domains similar to the target domain. From a collaborative perspective, Chapter 5 presents a feasible federated learning framework to achieve multi-source learning through the use of limited labelled data from different energy users. Extensive experimental studies validate that our approaches enable the established model to generalise well with different task users.

6.1 Future Work

We identify four research directions for the ongoing studies based on our prior works, which are provided as follows:

- Although it is empirically proved in Chapters 2 and 3 that the variants of GBDT show outstanding performance over wide resource-constrained devices for energy-related predictive analysis tasks, the prior works on parallelism and training acceleration are still limited on the tree-based ensemble algorithms, as discussed in Chapter 5. We plan to develop more cost-effective optimisation solutions for constructing gradient-boosting trees across various heterogeneous edge computing systems.
- In this thesis, the conventional deterministic point prediction, which is employed for PV forecasting and NILM tasks, is often error-prone, and bad predictions potentially jeopardise the online scheduling of energy resources. As a feasible alternative, distributional prediction is introduced as a feasible alternative to tame the detrimental energy-related estimates by jointly considering a specified confidence interval of predictions. In future work, we will explore cost-effective learning methods for distributional prediction. This approach will allow energy management systems to have better control over the uncertainties in decision making and significantly improve the power system's reliability and energy scheduling performance.
- The transfer learning techniques proposed in Chapter 3 and 4 help solve the data shortage issues and power NILM applications significantly. However, NILM modelling only requires main readings as the input features, which is relatively convenient to be collected. In other real-world applications, e.g. PV output forecasting and load prediction, the full dimensions of well-characterised features are only sometimes available in different domains, which makes knowledge transfer more difficult for multi-source learning tasks. To this end, we will develop a unified heterogeneous transfer learning paradigm (though feature dimensions and data distribution may be different) for state-of-the-art ML models. Our design aims to extract domain-invariant semantic information from multiple domains with varying spaces of features. A dimension-agnostic transferability quantification strategy needs to be

explored to effectively disclose the relationships between source and target domains in order to be able to fully use learned knowledge from available data sources.

- Regarding the collaborative modelling of boosting trees in Chapter 5, we first plan to use a wide range of single-board computers to replace the Dask Fargate cluster as the geographically distributed edge devices in our prototype system. Such implementation will provide us with a more sensitive testing environment to study our framework's effectiveness compared to GBDT-based approaches and other available alternatives [134, 119, 19]. Second, we will explore the performance of node-level parallelism on the leaf-wise model. [98] made the first attempt in a high-performance computing environment, which we aim to extend to broader application scenarios. Third, we will develop a location-aware decentralised training architecture for privacy-preserving collaborative learning. This will help to avoid costly communication overheads with the far-end server. Last, and most importantly, we will investigate the potential of the asynchronous training of GBDT [29], and integrate an asynchronous training strategy into the design of a federated learning system. Thus, our design will be more adaptable and able to be used in a complex distributed computing environment with heterogeneous devices.

Bibliography

[1] Solcast. https://solcast.com.au. Accessed: 2018-10-25.

[2] Mohamed Abdel-Nasser and Karar Mahmoud. Accurate photovoltaic power forecasting models using deep lstm-rnn. *Neural Computing and Applications*, 31(7):2727–2740, 2019.

[3] Ali Adabi, Pavlo Manovi, and Patrick Mantey. Seads: A modifiable platform for real time monitoring of residential appliance energy consumption. In *2015 Sixth International Green and Sustainable Computing Conference (IGSC)*, pages 1–4. IEEE, 2015.

[4] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 439–450, 2000.

[5] Awadelrahman MA Ahmed, Yan Zhang, and Frank Eliassen. Generative adversarial networks and transfer learning for non-intrusive load monitoring in smart grids. In *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–7. IEEE, 2020.

[6] Mohamed H Albadi and Ehab F El-Saadany. A summary of demand response in electricity markets. *Electric power systems research*, 78(11):1989–1996, 2008.

[7] Martin Anthony and Norman Biggs. Computational learning theory: an introduction, 1992.

[8] Charles R Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

[9] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[10] Nipun Batra, Amarjeet Singh, and Kamin Whitehouse. Gemello: Creating a detailed energy breakdown from just the monthly electricity bill. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 431–440, 2016.

[11] Gissella Bejarano, David DeFazio, and Arti Ramesh. Deep latent generative models for energy disaggregation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 850–857, 2019.

[12] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

[13] Tal Ben-Nun and Torsten Hoefler. Demystifying parallel and distributed deep learning: An in-depth concurrency analysis. *ACM Computing Surveys (CSUR)*, 52(4):1–43, 2019.

[14] Harisankar Bendu, BBVL Deepak, and S Murugan. Multi-objective optimization of ethanol fuelled hcci engine performance using hybrid grnn–pso. *Applied energy*, 187:601–611, 2017.

[15] Christian Blum and Xiaodong Li. Swarm intelligence in optimization. In *Swarm intelligence: introduction and applications*, pages 43–85. Springer, 2008.

[16] Roberto Bonfigli, Andrea Felicetti, Emanuele Principi, Marco Fagiani, Stefano Squartini, and Francesco Piazza. Denoising autoencoders for non-intrusive load monitoring: improvements and comparative evaluation. *Energy and Buildings*, 158:1461–1474, 2018.

[17] Roberto Bonfigli, Emanuele Principi, Marco Fagiani, Marco Severini, Stefano Squartini, and Francesco Piazza. Non-intrusive load monitoring by using active and reactive power in additive factorial hidden markov models. *Applied Energy*, 208:1590–1607, 2017.

[18] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, pages 13–16, New York, NY, USA, 2012. ACM.

[19] Hui Cao, Shubo Liu, Renfang Zhao, and Xingxing Xiong. Ifed: A novel federated learning framework for local differential privacy in power internet of things. *International Journal of Distributed Sensor Networks*, 16(5):1550147720919698, 2020.

[20] Linda Capuano. International energy outlook 2018 (ieo2018). *US Energy Information Administration (EIA): Washington, DC, USA*, 2018:21, 2018.

[21] Xiaomin Chang, Wei Li, Jin Ma, Ting Yang, and Albert Y Zomaya. Interpretable machine learning in sustainable edge computing: A case study of short-term photovoltaic power output prediction. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8981–8985. IEEE, 2020.

[22] Xiaomin Chang, Wei Li, Chunqiu Xia, Qiang Yang, Jin Ma, Ting Yang, and Albert Zomaya. Transferable tree-based ensemble model for non-intrusive load monitoring. *IEEE Transactions on Sustainable Computing*, 2022.

[23] Xiaomin Chang, Wei Li, and Albert Y Zomaya. A lightweight short-term photovoltaic power prediction for edge computing. *IEEE Transactions on Green Communications and Networking*, 2020.

[24] Changsong Chen, Shanxu Duan, Tong Cai, Bangyin Liu, and Gangwei Hu. Smart energy management system for optimal microgrid economic operation. *IET renewable power generation*, 5(3):258–267, 2011.

[25] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[26] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, et al. Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4):1–4, 2015.

[27] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, Dimitrios Papadopoulos, and Qiang Yang. Secureboost: A lossless federated learning framework. *IEEE Intelligent Systems*, 2021.

[28] Woo-Seok Choi, Matthew Tomei, Jose Rodrigo Sanchez Vicarte, Pavan Kumar Hanumolu, and Rakesh Kumar. Guaranteeing local differential privacy on ultra-low-power systems. In *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, pages 561–574. IEEE, 2018.

[29] Cheng Daning, Xia Fen, Li Shigang, and Zhang Yunquan. Asynch-sgbdt: Asynchronous parallel stochastic gradient boosting decision tree based on parameters server. *arXiv e-prints*, pages arXiv–1804, 2018.

[30] Utpal Kumar Das, Kok Soon Tey, Mehdi Seyedmahmoudian, Saad Mekhilef, Moh Yamani Idna Idris, Willem Van Deventer, Bend Horan, and Alex Stojcevski. Forecasting of photovoltaic power generation and model optimization: A review. *Renewable and Sustainable Energy Reviews*, 81:912–928, 2018.

[31] Aurélien Delfosse, Georges Hebrail, and Aimen Zerroug. Deep learning applied to nilm: is data augmentation worth for energy disaggregation? In *ECAI 2020*, pages 2972–2977. IOS Press, 2020.

[32] Ravinesh C Deo, Xiaohu Wen, and Feng Qi. A wavelet-coupled support vector machine model for forecasting global incident solar radiation using limited meteorological dataset. *Applied Energy*, 168:568–593, 2016.

[33] Mengnan Du, Ninghao Liu, and Xia Hu. Techniques for interpretable machine learning. *arXiv preprint arXiv:1808.00033*, 2018.

[34] Michele D'Incecco, Stefano Squartini, and Mingjun Zhong. Transfer learning for non-intrusive load monitoring. *IEEE Transactions on Smart Grid*, 11(2):1419–1429, 2019.

[35] Karen Ehrhardt-Martinez, Kat A Donnelly, Skip Laitner, et al. Advanced metering initiatives and residential feedback programs: a meta-review for household electricity-saving opportunities. American Council for an Energy-Efficient Economy Washington, DC, 2010.

[36] Wenjing Fang, Chaochao Chen, Bowen Song, Li Wang, Jun Zhou, and Kenny Q Zhu. Adapted tree boosting for transfer learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 741–750. IEEE, 2019.

[37] Anthony Faustine, Lucas Pereira, Hafsa Bousbiat, and Shridhar Kulkarni. Unet-nilm: A deep neural network for multi-tasks appliances state detection and power estimation in nilm. In *Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring*, pages 84–88, 2020.

[38] Corinna Fischer. Feedback on household electricity consumption: a tool for saving energy? *Energy efficiency*, 1(1):79–104, 2008.

[39] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

[40] Xile Gao, Haiyong Luo, Qu Wang, Fang Zhao, Langlang Ye, and Yuexia Zhang. A human activity recognition algorithm based on stacking denoising autoencoder and lightgbm. *Sensors*, 19(4):947, 2019.

[41] Ronald L Graham. Bounds for certain multiprocessing anomalies. *Bell system technical journal*, 45(9):1563–1581, 1966.

[42] Ronald Lewis Graham, Eugene Leighton Lawler, Jan Karel Lenstra, and AHG Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. In *Annals of discrete mathematics*, volume 5, pages 287–326. Elsevier, 1979.

[43] Marjorie G Hahn. Probability in banach spaces: Isoperimetry and processes., 1994.

[44] Nick Howgrave-Graham and Antoine Joux. New generic algorithms for hard knapsacks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 235–256. Springer, 2010.

[45] Anders Huss. Hybrid model approach to appliance load disaggregation: Expressive appliance modelling by combining convolutional neural networks and hidden semi markov models., 2015.

[46] Paris IEA. Electricity information: overview. *URL www. iea. org/reports/electricity-information-overview*, 2020.

[47] Ruoxi Jia, Yang Gao, and Costas J Spanos. A fully unsupervised non-intrusive load monitoring framework. In *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 872–878. IEEE, 2015.

[48] Sooyoung Jung and Yong Tae Yoon. Optimal operating schedule for energy storage system: Focusing on efficient energy management for microgrid. *Processes*, 7(2):80, 2019.

[49] Maria Kaselimi, Nikolaos Doulamis, Anastasios Doulamis, Athanasios Voulodimos, and Eftychios Protopapadakis. Bayesian-optimized bidirectional lstm regression model

for non-intrusive load monitoring. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2747–2751. IEEE, 2019.

[50] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pages 3146–3154, 2017.

[51] Jack Kelly and William Knottenbelt. Neural nilm: Deep neural networks applied to energy disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pages 55–64, 2015.

[52] Jack Kelly and William Knottenbelt. The uk-dale dataset, domestic appliance-level electricity demand and whole-house demand from five uk homes. *Scientific data*, 2(1):1–14, 2015.

[53] Mohammad Nazmul Alam Khan and Douglas R Heisterkamp. Adapting instance weights for unsupervised domain adaptation using quadratic mutual information and subspace learning. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 1560–1565. IEEE, 2016.

[54] J Zico Kolter and Matthew J Johnson. Redd: A public data set for energy disaggregation research. In *Workshop on data mining applications in sustainability (SIGKDD), San Diego, CA*, volume 25, pages 59–62, 2011.

[55] Weicong Kong, Zhao Yang Dong, Jin Ma, David J Hill, Junhua Zhao, and Fengji Luo. An extensible approach for non-intrusive load disaggregation with smart meter data. *IEEE Transactions on Smart Grid*, 9(4):3362–3372, 2016.

[56] Henning Lange and Mario Bergés. Efficient inference in dual-emission fhmm for energy disaggregation. In *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[57] Sonia Leva, Alberto Dolara, Francesco Grimaccia, Marco Mussetta, and Emanuele Ogliari. Analysis and validation of 24 hours ahead neural network forecasting of photovoltaic output power. *Mathematics and computers in simulation*, 131:88–100, 2017.

[58] Qinbin Li, Zeyi Wen, and Bingsheng He. Practical federated gradient boosting decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4642–4649, 2020.

[59] W. Li, T. Yang, F. C. Delicato, P. F. Pires, Z. Tari, S. U. Khan, and A. Y. Zomaya. On enabling sustainable edge computing with renewable energy resources. *IEEE Communications Magazine*, 56(5):94–101, May 2018.

[60] Wei Li, Xiaomin Chang, Junwei Cao, Ting Yang, Yaojie Sun, and Albert Y Zomaya. A sustainable and user-behavior-aware cyber-physical system for home energy management. *ACM Transactions on Cyber-Physical Systems*, 3(4):1–24, 2019.

[61] Wei Li, Igor Santos, Flavia C. Delicato, Paulo F. Pires, Luci Pirmez, Wei Wei, Houbing Song, Albert Zomaya, and Samee Khan. System modelling and performance evaluation of a three-tier cloud of things. *Future Generation Computer Systems*, 70:104 – 125, 2017.

[62] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.

[63] Jun Lin, Jin Ma, Jianguo Zhu, and Huishi Liang. Deep domain adaptation for non-intrusive load monitoring based on a knowledge transfer learning network. *IEEE Transactions on Smart Grid*, 13(1):280–292, 2021.

[64] Chang Liu, Lichen Wang, and Yun Fu. Meta adversarial weight for unsupervised domain adaptation. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pages 10–18. SIAM, 2022.

[65] Junhong Liu and Jouni Lampinen. A fuzzy adaptive differential evolution algorithm. *Soft Computing*, 9:448–462, 2005.

[66] Luyao Liu, Yi Zhao, Dongliang Chang, Jiyang Xie, Zhanyu Ma, Qie Sun, Hongyi Yin, and Ronald Wennersten. Prediction of short-term pv power output and uncertainty analysis. *Applied energy*, 228:700–711, 2018.

[67] Xu Liu, Yingguang Li, Qinglu Meng, and Gengxiang Chen. Deep transfer learning for conditional shift in regression. *Knowledge-Based Systems*, 227:107216, 2021.

[68] Yang Liu, Zhuo Ma, Ximeng Liu, Siqi Ma, Surya Nepal, Robert H Deng, and Kui Ren. Boosting privately: Federated extreme gradient boosting for mobile crowdsensing. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, pages 1–11. IEEE, 2020.

[69] Yinyan Liu, Li Zhong, Jing Qiu, Junda Lu, and Wei Wang. Unsupervised domain adaptation for non-intrusive load monitoring via adversarial and joint adaptation network. *IEEE Transactions on Industrial Informatics*, 2021.

[70] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015.

[71] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *International conference on machine learning*, pages 2208–2217. PMLR, 2017.

[72] Renzhi Lu, Seung Ho Hong, and Mengmeng Yu. Demand response for home energy management using reinforcement learning and artificial neural network. *IEEE Transactions on Smart Grid*, 10(6):6629–6639, 2019.

[73] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. Explainable ai for trees: From local explanations to global understanding. *arXiv preprint arXiv:1905.04610*, 2019.

[74] Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

[75] Zhanyu Ma, Jiyang Xie, Hailong Li, Qie Sun, Zhongwei Si, Jianhua Zhang, and Jun Guo. The role of data analysis in the development of intelligent energy networks. *IEEE Network*, 31(5):88–95, 2017.

[76] Stephen Makonin, Fred Popowich, Ivan V Bajić, Bob Gill, and Lyn Bartram. Exploiting hmm sparsity to perform online real-time nonintrusive load monitoring. *IEEE Transactions on smart grid*, 7(6):2575–2585, 2015.

[77] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

[78] Lukas Mauch, Karim Said Barsim, and Bin Yang. How well can hmm model load signals. In *Proceeding of the 3rd international workshop on non-intrusive load monitoring (NILM 2016)*, number 6, 2016.

[79] Lukas Mauch and Bin Yang. A novel dnn-hmm-based approach for extracting single loads from aggregate power signals. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2384–2388. IEEE, 2016.

[80] Stephen McLaughlin, Patrick McDaniel, and William Aiello. Protecting consumer privacy from electric load monitoring. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 87–98, 2011.

[81] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[82] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging.(2016). *arXiv preprint arXiv:1602.05629*, 2016.

[83] Tamer F Megahed, Sobhy M Abdelkader, and Ahmad Zakaria. Energy management in zero-energy building using neural network predictive control. *IEEE Internet of Things Journal*, 6(3):5336–5344, 2019.

[84] Adel Mellit and Alessandro Massi Pavan. A 24-h forecast of solar irradiance using artificial neural network: Application for performance prediction of a grid-connected pv plant at trieste, italy. *Solar Energy*, 84(5):807 – 821, 2010.

[85] Qi Meng, Guolin Ke, Taifeng Wang, Wei Chen, Qiwei Ye, Zhi-Ming Ma, and Tie-Yan Liu. A communication-efficient parallel algorithm for decision tree. *arXiv preprint arXiv:1611.01276*, 2016.

[86] Lilyana Mihalkova and Raymond J Mooney. Transfer learning by mapping with minimal target data. In *Proceedings of the AAAI-08 workshop on transfer learning for complex tasks*, 2008.

[87] David Murray, Lina Stankovic, and Vladimir Stankovic. An electrical load measurements dataset of united kingdom households from a two-year longitudinal study. *Scientific data*, 4(1):1–12, 2017.

[88] David Murray, Lina Stankovic, and Vladimir Stankovic. Explainable nilm networks. In *5th International Workshop on Non Intrusive Load Monitoring*, 2020.

[89] Gábor I Nagy, Gergő Barta, Sándor Kazi, Gyula Borbély, and Gábor Simon. Gefcom2014: Probabilistic solar and wind power forecasting using a generalized additive tree ensemble approach. *International Journal of Forecasting*, 32(3):1087–1093, 2016.

[90] B Neenan, J Robinson, and RN Boisvert. Residential electricity use feedback: A research synthesis and economic framework. *Electric Power Research Institute*, 3:123–129, 2009.

[91] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[92] Yungang Pan, Ke Liu, Zhaoyan Shen, Xiaojun Cai, and Zhiping Jia. Sequence-to-subsequence learning with conditional gan for power disaggregation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3202–3206. IEEE, 2020.

[93] Oliver Parson, Siddhartha Ghosh, Mark Weal, and Alex Rogers. Non-intrusive load monitoring using prior models of general appliance types. 2012.

[94] Prakash Pawar, Mudige TarunKumar, et al. An iot based intelligent smart energy management system with accurate forecasting and load strategy for renewable generation. *Measurement*, 152:107187, 2020.

[95] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[96] Hugo TC Pedro and Carlos FM Coimbra. Assessment of forecasting techniques for solar power production with no exogenous inputs. *Solar Energy*, 86(7):2017–2028, 2012.

[97] Martin Pelikan, David E Goldberg, and Erick Cantú-Paz. Boa: The bayesian optimization algorithm. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1*, pages 525–532. Morgan Kaufmann Publishers Inc., 1999.

[98] Bo Peng, Langshi Chen, Jiayu Li, Miao Jiang, Selahattin Akkas, Egor Smirnov, Ruslan Israfilov, Sergey Khekhnev, Andrey Nikolaev, and Judy Qiu. Harpgbdt: Optimizing gradient boosting decision tree for parallel efficiency. In *2019 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 1–11. IEEE, 2019.

[99] Fernando Pérez-Cruz. Kullback-leibler divergence estimation of continuous distributions. In *2008 IEEE international symposium on information theory*, pages 1666–1670. IEEE, 2008.

[100] Dario Piga, Andrea Cominola, Matteo Giuliani, Andrea Castelletti, and Andrea Emilio Rizzoli. Sparse optimization for automated energy end use disaggregation. *IEEE Transactions on Control Systems Technology*, 24(3):1044–1051, 2015.

[101] Hasan Rafiq, Xiaohan Shi, Hengxu Zhang, Huimin Li, Manesh K Ochani, and Aamer A Shah. Generalizability improvement of deep learning based non-intrusive load monitoring system using data augmentation. *IEEE Transactions on Smart Grid*, 2021.

[102] M Mazhar Rathore, Anand Paul, Won-Hwa Hong, HyunCheol Seo, Imtiaz Awan, and Sharjil Saeed. Exploiting iot and big data analytics: Defining smart digital city using real-time urban data. *Sustainable cities and society*, 40:600–610, 2018.

[103] Andreas Reinhardt and Mazen Bouchur. On the impact of the sequence length on sequence-to-sequence and sequence-to-point learning for nilm. In *Proceedings of the 5th International Workshop on Non-Intrusive Load Monitoring*, pages 75–78, 2020.

[104] Guillaume Richard, Antoine de Mathelin, Georges Hébrail, Mathilde Mougeot, and Nicolas Vayatis. Unsupervised multi-source domain adaptation for regression. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 395–411. Springer, 2020.

[105] Greg Ridgeway. Generalized boosted models: A guide to the gbm package. *Update*, 1(1):2007, 2007.

[106] Janessa Rivera and Rob van der Meulen. Gartner says the internet of things installed base will grow to 26 billion units by 2020. *Stamford, conn., December*, 12, 2013.

[107] Matthew Rocklin. Dask: Parallel computation with blocked algorithms and task scheduling. In *Proceedings of the 14th python in science conference*, volume 130, page 136. Citeseer, 2015.

[108] Hanmin Sheng, Jian Xiao, Yuhua Cheng, Qiang Ni, and Song Wang. Short-term solar power forecasting based on weighted gaussian process regression. *IEEE Transactions on Industrial Electronics*, 65(1):300–308, 2017.

[109] Wenbo Shi, Na Li, Chi-Cheng Chu, and Rajit Gadh. Real-time energy management in microgrids. *IEEE Transactions on Smart Grid*, 8(1):228–238, 2015.

[110] Yunchuan Shi, Wei Li, Xiaomin Chang, and Albert Y Zomaya. User privacy leakages from federated learning in nilm applications. In *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 212–213, 2021.

[111] Changho Shin, Sunghwan Joo, Jaeryun Yim, Hyoseop Lee, Taesup Moon, and Wonjong Rhee. Subtask gated networks for non-intrusive load monitoring. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1150–1157, 2019.

[112] Le Song, Jonathan Huang, Alex Smola, and Kenji Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 961–968, 2009.

[113] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[114] Lei Sun, Zhenzhi Lin, Yan Xu, Fushuan Wen, Can Zhang, and Yusheng Xue. Optimal skeleton-network restoration considering generator start-up sequence and load pickup. *IEEE Transactions on Smart Grid*, 10(3):3174–3185, 2018.

[115] Zhihua Tian, Rui Zhang, Xiaoyang Hou, Jian Liu, and Kui Ren. Federboost: Private federated learning for gbdt. *arXiv preprint arXiv:2011.02796*, 2020.

[116] Stacey Truex, Ling Liu, Ka-Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. Ldp-fed: Federated learning with local differential privacy. In *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, pages 61–66, 2020.

[117] Sanna Tuomela, Mauricio de Castro Tomé, Netta Iivari, and Rauli Svento. Impacts of home energy management systems on electricity consumption. *Applied Energy*, 299:117310, 2021.

[118] Jurgen Van Gael. *Bayesian Nonparametric Hidden Markov Models*. PhD thesis, Citeseer, 2012.

[119] Haijin Wang, Caomingzhe Si, Junhua Zhao, Guolong Liu, and Fushuan Wen. Fednilm: A federated learning-based non-intrusive load monitoring method for privacy-protection. *arXiv preprint arXiv:2105.11085*, 2021.

[120] Haoxiang Wang, Jiasheng Zhang, Chenbei Lu, and Chenye Wu. Privacy preserving in non-intrusive load monitoring: A differential privacy perspective. *IEEE Transactions on Smart Grid*, 12(3):2529–2543, 2020.

[121] Zheng Wang, Irena Koprinska, and Mashud Rana. Solar power prediction using weather type pair patterns. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 4259–4266. IEEE, 2017.

[122] Zheng Wang, Irena Koprinska, Alicia Troncoso, and Francisco Martínez-Álvarez. Static and dynamic ensembles of neural networks for solar power forecasting. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[123] Yuncheng Wu, Shaofeng Cai, Xiaokui Xiao, Gang Chen, and Beng Chin Ooi. Privacy preserving vertical federated learning for tree-based models. *arXiv preprint arXiv:2008.06170*, 2020.

[124] Chunqiu Xia, Wei Li, Xiaomin Chang, Flavia Delicato, Ting Yang, and Albert Zomaya. Edge-based energy management for smart homes. In *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*, pages 849–856. IEEE, 2018.

[125] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2272–2281, 2017.

[126] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[127] M Zamo, O Mestre, P Arbogast, and O Pannekoucke. A benchmark of statistical regression methods for short-term forecasting of photovoltaic electricity production, part i: Deterministic forecast of hourly production. *Solar Energy*, 105:792–803, 2014.

[128] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. 2016.

[129] Chaoyun Zhang, Mingjun Zhong, Zongzuo Wang, Nigel Goddard, and Charles Sutton. Sequence-to-point learning with neural networks for nonintrusive load monitoring. *arXiv preprint arXiv:1612.09106*, 2016.

[130] Chaoyun Zhang, Mingjun Zhong, Zongzuo Wang, Nigel Goddard, and Charles Sutton. Sequence-to-point learning with neural networks for non-intrusive load monitoring. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[131] Chi Zhang, Sanmukh R Kuppannagari, Rajgopal Kannan, and Viktor K Prasanna. Building hvac scheduling using reinforcement learning via neural network based model approximation. In *Proceedings of the 6th ACM international conference on systems for energy-efficient buildings, cities, and transportation*, pages 287–296, 2019.

[132] Shirong Zhang and Yuling Tang. Optimal schedule of grid-connected residential pv generation systems with battery storages under time-of-use and step tariffs. *Journal of Energy Storage*, 23:175–182, 2019.

[133] Wenying Zhang, Huaguang Zhang, Jinhai Liu, Kai Li, Dongsheng Yang, and Hui Tian. Weather prediction with multiclass support vector machines in the fault detection of photovoltaic system. *IEEE/CAA Journal of Automatica Sinica*, 4(3):520–525, 2017.

[134] Yu Zhang, Guoming Tang, Qianyi Huang, Yi Wang, Kui Wu, Keping Yu, and Xun Shao. Fednilm: Applying federated learning to nilm applications at the edge. *IEEE Transactions on Green Communications and Networking*, 2022.

[135] Han Zhao, Shanghang Zhang, Guanhang Wu, Geoffrey J Gordon, et al. Multiple source domain adaptation with adversarial learning. 2018.

[136] Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. Adversarial multiple source domain adaptation. *Advances in neural information processing systems*, 31, 2018.

[137] Lingchen Zhao, Lihao Ni, Shengshan Hu, Yaniiao Chen, Pan Zhou, Fu Xiao, and Libing Wu. Inprivate digging: Enabling tree-based distributed data mining with differential privacy. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 2087–2095. IEEE, 2018.

[138] Bin Zhou, Wentao Li, Ka Wing Chan, Yijia Cao, Yonghong Kuang, Xi Liu, and Xiong Wang. Smart home energy management systems: Concept, configurations, and scheduling strategies. *Renewable and Sustainable Energy Reviews*, 61:30–40, 2016.

[139] Zejian Zhou, Yingmeng Xiang, Hao Xu, Zhehan Yi, Di Shi, and Zhiwei Wang. A novel transfer learning-based intelligent nonintrusive load-monitoring with limited measurements. *IEEE Transactions on Instrumentation and Measurement*, 70:1–8, 2020.