A Distributed and Real-time Machine Learning Framework for Smart Meter Big Data

Shuang Dai

A thesis submitted for the degree of *Doctor of Philosophy* in Data Science

Department of Mathematical Sciences
University of Essex

January 2023

Abstract

The advanced metering infrastructure allows smart meters to collect high-resolution consumption data, thereby enabling consumers and utilities to understand their energy usage at different levels, which has led to numerous smart grid applications. Smart meter data, however, poses different challenges to developing machine learning frameworks than classic theoretical frameworks due to their big data features and privacy limitations.

Therefore, in this work, we aim to address the challenges of building machine learning frameworks for smart meter big data. Specifically, our work includes three parts: 1) We first analyze and compare different learning algorithms for multi-level smart meter big data. A daily activity pattern recognition model has been developed based on non-intrusive load monitoring for appliance-level smart meter data. Then, a consensus-based load profiling and forecasting system has been proposed for individual building level and higher aggregated level smart meter data analysis; 2) Following discussion of multi-level smart meter data analysis from an offline perspective, a universal online functional analysis model has been proposed for multi-level real-time smart meter big data analysis. The proposed model consists of a multi-scale load dynamic profiling unit based on functional clustering and a multi-scale online load forecasting unit based on functional deep neural networks. The two units enable online tracking of the dynamic cluster trajectories and online forecasting of daily multi-scale demand; 3) To enable smart meter data analysis in the distributed environment, *FederatedNILM* was proposed, which is then combined with differential privacy to provide privacy guarantees for the appliance-level distributed machine learning framework. Based on federated deep learning enhanced with two schemes, namely the utility optimization scheme and the privacy-preserving scheme, the proposed distributed and privacy-preserving machine learning framework enables electric utilities and service providers to offer smart meter services on a large scale.

Nomenclature

ADLs Activities of daily life

AMI Advanced metering infrastructure

ANN Artificial neural network

ARMA Auto-regressive moving-average

ARMIA Auto-regressive integrated moving average

BIC Bayesian information criterion

BP Backpropagation

CH Calinski-Harabasz

CNN Convolutional neural network

CO Combinatorial optimization

CVI Cluster validity indice

DB-index Davies-Buldin validity index

DNN Deep neural network

DPFL Differential private federated learning

DTW Dynamic time warping distance

FDA Functional data analysis

FDN Functional deep neural network

FHMM factorial hidden Markov model

FL Federated learning

FN False negative

FP False positive

FPCA Functional principal component analysis

FR Functional regression

FTL Federated transfer learning

GB Gradient boosting

GDPFL Global differential private federated learning

GDPR General data protection regulation

GMM Gaussian mixture model

HFL Horizontal federated learning

HMM Hidden Markov model

HR Hit rate

IoT Internet of things

IR Improvement rate

KNR K-nearest regression

LDPFL Local differential private federated learning

LSTM Long short-term memory

MA Moving average

MAE Mean absolute error

MAPE Mean absolute percentage error

MFPCA Multivariate functional principal components analysis

MLR Multiple linear regression

MSE Mean squared error

NILM Non-intrusive load monitoring

OL Online learning

PA Passive-aggressive

PCA Principal component analysis

PSP-Net Pyramid scene parsing network

RBF Radial basis function

RBFN Radial basis function network

RNNs Recurrent neural networks

SAE Signal aggregated error

SBD Shape-based distance

SGD Stochastic gradient descent

SV Support vector

SVM Support vector machines

SVR Support vector regression

TL Transfer learning

TP True positive

VFL Vertical federated learning

XGBoost Extreme gradient boosting

Declaration of Authorship

No portion of the work presented in this thesis has been submitted as part of an application for any other degree or qualification at this or any other institution of learning.

Acknowledgements

My first and foremost thanks go to my supervisor, Dr. Fanlin Meng, for his continued mentoring and foresight throughout this project. The regular meetings continually motivated me to think creatively and shaped the project constructively.

I would like to thank my fiancé Jiawei Hu for his constant support during the arduous time of writing my thesis. Also, I would like to thank my family for always being interested in my project stories and supporting my new ideas. Special thanks to Dr. Yang Zhang for accompanying me through the final version of this thesis.

Finally, I would like to thank my supervisor, Dr. Yanchun Bao, for her help during the completion of this Ph.D. project.

Contents

Abstract i
Declaration of Authorship v
Acknowledgements vi

1 Introduction **1**
1.1 Context and Problem Statement 1
1.2 Research Aims and Objectives 2
1.3 Contributions 3
1.4 Thesis Outline 4
1.5 List of Publications 4

2 Background and Literature Review **6**
2.1 Load Analysis 6
2.1.1 Load Profiling 8
2.1.2 Load Forecasting 9
2.2 Multi-level Load Analysis with Smart Meter Data 17
2.2.1 Appliance Level Load Forecasting 18
2.2.2 Individual Level Load Forecasting 20
2.2.3 Higher Aggregation Level Load Forecasting 21
2.3 Enhancing Techniques for Load Analysis 22
2.3.1 Transfer Learning 22
2.3.2 Online Learning 24
2.3.3 Distributed and Privacy-preserving Machine Learning 25
2.3.4 Discussion 28
2.4 Chapter Summary 29

3 Load Profiling and Forecasting System for Multi-Level Smart Meter Data Analysis **30**
3.1 Introduction 30

3.2 Problem Statement 31
3.3 Appliance Level load Analysis: An ADLs Patterns Recognition NILM Model 34
3.3.1 Preliminaries 34
3.3.2 Framework Design Overview 35
3.3.3 Implementation 36
3.3.4 Evaluation and Discussion 39
3.4 Individual Level and Higher Aggregation Level load Forecasting: A Consensus-based load Profiling and Forecasting System 47
3.4.1 Framework Design Overview 47
3.4.2 Implementation 49
3.4.3 Evaluation and Discussion 62
3.5 Chapter Summary 80

4 From Offline to Online: Real-time Smart Meter Data Analysis 81
4.1 Introduction 81
4.2 Problem Statement 82
4.3 A Universal Online Functional Analysis Model for Multi-scale Load Dynamic Profiling and Forecasting 83
4.3.1 Framework Design Overview 83
4.3.2 Implementation 86
4.3.3 Evaluation and Discussion 93
4.3.4 Summary 105
4.4 Chapter Summary 105

5 Distributed and Privacy-preserving Machine Learning Framework for Smart Meter Data Applications 107
5.1 Introduction 107
5.2 Problem Statement 108
5.3 Preliminaries 109
5.3.1 Federated deep learning 109
5.3.2 Differential privacy 110
5.4 FederatedNILM: A Distributed Smart Meter Analysis Framework . . 111
5.4.1 Framework Design and Implementation 111
5.4.2 Evaluation and Discussion 115
5.5 DP^2-NILM: Providing Privacy Guarantee to Distributed Smart Meter Analysis 124
5.5.1 Framework Design Overview 124
5.5.2 Utility Optimization of DP^2-NILM 128

5.5.3 Privacy-preserving of DP^2-NILM 130
5.5.4 Evaluation and Discussion 131
5.6 Chapter Summary 139

6 Conclusion and Future Directions **141**
6.1 Summary of Contributions 141
6.1.1 Load Profiling and Forecasting System for Multi-Level Smart Meter Data Analysis 141
6.1.2 From Offline to Online: Real-time Smart Meter Data Analysis 142
6.1.3 Distributed and Privacy-preserving Machine Learning Framework for Smart Meter Data Applications 143
6.2 Future Directions 144

Bibliography **171**

Word Count: 33445

List of Figures

2.1 The general process of load analysis. 7
2.2 The general process of load clustering. 8
2.3 The general process of load forecasting. 10
2.4 An example structure of ANN. 13
2.5 The calculation process of a neuron in ANN. 13
2.6 Daily smart meter readings. 18
2.7 Transfer learning process. 23

3.1 The proposed ADLs pattern recognition NILM framework. 35
3.2 ADLs pattern recognition model implementation. 36
3.3 Improved NILM model based on sequence-to-point and transfer learning. 38
3.4 Load disaggregation results for kettle 42
3.5 Load disaggregation results for fridge 42
3.6 Load disaggregation results for dishwasher 43
3.7 Load disaggregation results for microwave 43
3.8 Load disaggregation results for washing machine 43
3.9 Usage frequency comparison of kettle, dishwasher and microwave . . 45
3.10 ADLs pattern recognition for dishwasher and microwave: cluster 1 represents the OFF status and cluster 2 represents the ON status . . 46
3.11 The framework of the consensus-based load profiling and forecasting system. 50
3.12 Load consumption of different types of buildings in four seasons. . . 51
3.13 Load consumption of different number of floors in four seasons. . . . 52
3.14 Load consumption of different heating types of buildings in four seasons. 53
3.15 Load consumption of different building energy rating certificate in four seasons. 54
3.16 Dynamic analysis to capture seasonal change. 58

3.17 Consensus-based model training process at the building level and the aggregation level. 59
3.18 A comparison of the CVIs of consensus-based clustering in different seasons. The grey vertical dashes line indicates the optimal cluster number for each CVI. 63
3.19 The cluster centroids of clusters in different seasons. 65

4.1 The proposed Universal-OFA model. 85
4.2 An example of the true daily load readings compared with the smoothed functional curves. 88
4.3 The architecture of the proposed FDN. 89
4.4 Multi-scale online FDN model updating process. 91
4.5 Functional clusters universal-sketching results. The red dashed vertical lines denote the peak period (17:00–18:30), the blue dashed vertical lines denote the daytime period (8:00–16:30 and 19:00–23:00), and the black dashed vertical lines denote the overnight period (from 23:00 to 07:30). 95
4.6 A dynamic cluster trajectory example at the individual levels (upper) and the dynamic cluster trajectory at the region level (lower). 100
4.7 MAE scores for forecasted daily load of 100 days of different models. 102
4.8 Box plot of the MAE scores for 48 half-hourly forecasted daily load of 100 unseen days of the Individual-OFA and Individual-OFA$_{offline}$. The black short lines are the daily median MAE and the red dots are the daily mean MAE. 105

5.1 Training process of the federated deep learning framework. 110
5.2 The overall layout of the deep learning model for NILM 114
5.3 Disaggregation performance on the test dataset in the seen house case. 119
5.4 Disaggregation performance in the unseen house case. 120
5.5 Training time for the proposed FederatedNILM and the centralized NILM model 122
5.6 The workflow of proposed DP^2-NILM framework. 125
5.7 The overall layout of the deep learning model for NILM. 127
5.8 Example fridge usage distributions for UK-DALE, REDD, and REFIT. 136
5.9 The ASRs of FedAvg-NILM, the GDPFL-NILM, and the LDPFL-NILM in the DP^2-NILM framework. 139

6.1 A vision of online FTL framework 146

List of Tables

2.1 Associated key stakeholders and benefits of load forecasting 10
2.2 Implementation scenarios of enhancing techniques 28

3.1 Summary of focused challenges, applications and advantages of different load analysis levels 32
3.2 Number of parameters, training time, and saving percentages of the proposed model compared to the original model 41
3.3 Test results for the proposed model and original model; Best results are marked in bold 44
3.4 Hit rate for the target appliances 46
3.5 Physical information of the considered 169 buildings. 48
3.6 Frequency analysis of the physical information for the simulated offices. 55
3.7 List of the used combination of the consensus-based clustering algorithm. DTW: dynamic time warping, SBD: shape-based distance; PAM: partition around medoids; DBA: DTW barycenter averaging. 57
3.8 Lists of the optimal clustering algorithms and their corresponding CVIs for the four seasons. 64
3.9 Chi-square test results for spring clusters. 66
3.10 Chi-square test results for summer clusters. 67
3.11 Chi-square test results for autumn clusters. 68
3.12 Chi-square test results for winter clusters. 69
3.13 Clusters profile summary for the four seasons. 70
3.14 Category summary for the dynamic cluster trajectories over the year. DT: dynamic trajectory. 71
3.15 Selected features for hourly load forecasting 73
3.16 First 10 most important features in Spring 74
3.17 First 10 most important features in Summer 75
3.18 First 10 most important features in Autumn 75

3.19 First 10 most important features in Winter 76
3.20 First 10 most important features for the 169 aggregated buildings . . 77
3.21 Optimal parameters for each forecast model 77
3.22 Best Parameters for each model 78
3.23 Forecasting results for each model 79

4.1 The Time-of-Use tariff. 86
4.2 The number of selected participants. 93
4.3 BIC scores 94
4.4 The social information sketch and the IVSs of the universally-sketched clusters. 96
4.5 Refined social information sketch 98
4.6 Evaluation mean scores and the improve rate of 100 days for different models 101
4.7 The MAE score of each cluster in different forecasting scenarios . . . 104

5.1 Relevant thresholds information 116
5.2 Training, validating and testing splits for the seen house case model 117
5.3 Training and testing splits for the unseen house case model 117
5.4 Parameters used in FederatedNILM 118
5.5 Test results for fridge 121
5.6 Test results for dishwasher 121
5.7 Test results for washing machine 122
5.8 Training and testing houses for the unseen house case model 123
5.9 Comparison results of Federated-NILM with state-of-the-arts 124
5.10 Distribution of the selected datasets. D: Days 132
5.11 Relevant threshold information 133
5.12 Parameters used in the DP^2-NILM framework 134
5.13 Average performance scores of the Local-NILM models, the Centralized-NILM model, and the FL-setting of DP^2-NILM for 9 households . . 135
5.14 Average performance scores of FedAvg-NILM and FedProx-NILM schemes for 9 households 137
5.15 Average performance scores of the GDPFL-NILM and the LDPFL-NILM schemes for 9 households 138

Chapter 1

Introduction

1.1 Context and Problem Statement

With the rollout of advanced metering infrastructure (AMI) [243], the power system is undergoing rapid evolution. The number of smart meters operating in UK households reached 27.5 million by the middle of 2022 [2], and the installation of smart meters through the real-time exchange of information between power suppliers and end-users will not only improve the efficiency of demand management but will also introduce numerous new load forecasting strategies [168]. The ability to record high-resolution consumption data allows smart meters to provide consumers and utilities with a deeper understanding of their energy consumption at different levels, allowing smart grids to be used in various ways. An accurate analysis of electricity consumption patterns makes it possible to derive a model for similar electricity consumption patterns, thereby providing a solid foundation for load forecasting, tariff adjustment, and customizing consumer preferences.

On the other hand, recent advancements in machine learning have propelled the broad utilization of smart technologies, particularly the internet of things (IoT). Worldwide, the number of IoT devices is expected to nearly triple from 8.74 billion in 2020 to more than 25 billion in 2030 [90]. The massive data collected from IoT devices is considered critical for constructing robust machine learning models, which has created a wealth of chances for growing innovations in the era of big data. However, real-world machine learning achievements have relied on the availability of vast amounts of well-labelled data, such as ImgNet [49] and Alpha Zero [88], which can be prohibitively expensive, particularly in fields requiring expertise and human skill. Moreover, IoT big data, characterized by high volume, high velocity, and high diversity [138], cannot be utilized directly as high-quality ready inputs,

posing many obstacles to the development of current data-driven real-world machine learning systems.

The challenges of developing a distributed and real-time machine learning framework in the era of smart meter big data are distinct from those of classic theoretical frameworks, owing to the features of big data and the restrictions placed by new data regulations and laws. These distinctions have essential effects on the assumptions and performance measures underlying the design of such a system. They may stimulate the development of more innovative and practical machine learning algorithms. There are several general challenges associated with analysing smart meter big data:

1) Most existing research focuses on forecasting aggregated smart meter data, ignoring load trends occurring at lower aggregation levels, such as the individual building/household level or the appliance level. These studies can impede the development of more accurate demand management strategies, as they obscure the visibility of embedding more personalized and refined models.

2) Smart meters realize the real-time acquisition of electricity consumption data, which poses difficulties for conventional offline machine learning frameworks that rely on historical smart meter readings, and necessitates a dynamic machine learning framework capable of handling the fine-grained timely data.

3) Smart meters are generally distributed in individual households rather than a centralized location, and the data acquisition process takes remote or local communication channels as carriers to complete the hierarchical transmission between the systems [119], and these data are often statistical disparities.

4) The hierarchical transmission of smart meter data has become increasingly challenging from a legislative standpoint. For example, the European Union's new General Data Protection Regulation (GDPR) [156] has several provisions safeguarding user privacy and restricting companies from transferring data without explicit user consent.

Therefore, it is necessary to develop a more intelligent and robust framework for analysing the smart meter big data to address these challenges.

1.2 Research Aims and Objectives

This research aims to develop and refine machine learning frameworks for smart meter analysis in distributed and real-time scenarios through innovative approaches. Toward achieving this aim, the following objectives were established:

- To address challenge 1): employee advanced deep learning techniques to enhance smart meter data analysis at the appliance level, and develop a robust machine learning framework to analyze smart meter data at the building and aggregate levels. Furthermore, examine the efficacy of advanced learning algorithms to explore the hidden relationship between different algorithms and varying levels of smart meter-based load usage patterns.

- To address challenge 2): establish a dynamic smart meter-based load analysis model that can reflect the change in consumers' electricity consumption behaviour, with the goal of real-time updating of electricity consumption data. In this process, dynamic clustering and online forecasting are combined to constantly cluster and forecast the demand of customers at different data aggregation levels.

- To address challenges 3) and 4): develop smart meter-based load forecasting systems, and utilise distributed machine learning mechanisms and privacy-preserving techniques to improve the efficiency of smart meter big data processing while protecting customer privacy.

1.3 Contributions

Three contributions have been made to this thesis, which can be summarized as follows:

- To enhance multi-level smart meter data analysis, a novel ADLs pattern recognition model was proposed to detect the smart meter-based load usage patterns at the appliance level. The model combines deep neural networks with advanced data analysis tools, including principle components analysis and k-means clustering. It helps to infer the daily routine of customers by forecasting appliance usage patterns. Then, a robust machine learning framework was developed to examine the efficacy of various load forecasting models at various data aggregation levels, such as building and aggregated group levels. As part of the developed framework, customized forecasting models are developed, and relationships between building characteristics and model forecasting performance are also analysed.

- A two-unit universal online functional analysis model (Universal-OFA) that has universal applicability for dynamically profiling and forecasting multi-scale demand was proposed. The Universal-OFA incorporates adaptive clustering

to identify dynamic load usage trajectories and employs real-time smart meter feedback to predict daily load requirements. The proposed Universal-OFA model has demonstrated its superiority based on real-world data to assist dynamic pricing strategies and customized electricity consumption management at multiple scales.

- A distributed federated deep learning framework for non-intrusive load monitoring (FederatedNILM) was first proposed, which combines federated learning with a state-of-the-art deep learning architecture to conduct smart meter-based load analysis. Then, we make the first attempt to conduct FL-based NILM focusing on utility optimization and privacy-preserving by developing a distributed and privacy-preserving NILM (DP^2-NILM) framework and carrying out comparative experiments on practical NILM scenarios based on real-world smart meter datasets. Extensive comparison experiments are conducted on three real-world datasets to evaluate the proposed framework.

1.4 Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 first introduces the background and reviews the related state-of-the-art of load analysis. It explores the general process of load forecasting, multi-level load forecasting, and enhancing techniques for load forecasting. Chapter 3 then proposes a daily activity recognition NILM model for appliance level smart meter data analysis and a consensus-based load profiling and forecasting system for building level and aggregation level smart meter analysis. After this, Chapter 4 expands the smart meter analysis system to the online environment, in which a two-unit universal online functional analysis model with universal applicability for dynamic profiling and forecasting multi-scale load is proposed. Chapter 5 then extends the appliance level smart meter analysis framework into a distributed and privacy-preserving environment by integrating federated learning and differential privacy mechanisms. Lastly, Chapter 6 concludes this work and identifies future research directions.

1.5 List of Publications

- Journal Papers

    1. Dai, S., & Meng, F. (2022). Addressing modern and practical challenges in Machine Learning: A Survey of online federated and transfer learning. *Applied Intelligence*.

2. Dai, S., Meng, F., Wang, Q., & Chen, X. (2022). DP ^{2}-NILM: A Distributed and Privacy-preserving Framework for Non-intrusive Load Monitoring. *Renewable and Sustainable Energy Reviews* (in Revision).
3. Dai, S., Meng, F., Dai, H., Wang, Q., & Chen, X. (2021). Electrical peak demand forecasting-A review. *Renewable and Sustainable Energy Reviews* (in Review Process).
4. Chen, H., Dai, S., Xiao, B., & Meng, F. (2022). Smart building indoor thermal profiling system: A data-driven approach based on consensus and dynamic clustering *IEEE Transactions on Engineering Management* (in Review Process).
5. Dai, S., & Meng, F. (2022). A Universal Online Functional Analysis Model for Multi-scale Load Demand Dynamic Profiling and Forecasting. *Technological Forecasting and Social Change* (in Preparation).

- Conference Proceedings

5. Dai, S., & Meng, F. (2020, July). Energy Forecasting with Building Characteristics Analysis. In *2020 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-7). IEEE.
6. Dai, S., Wang, Q., & Meng, F. (2021, July). A telehealth framework for dementia care: an ADLs patterns recognition model for patients based on NILM. In *2021 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.
7. Dai, S., & Meng, F. (2022, September). Online Model-based Functional Clustering and Functional Deep Learning for Load Forecasting Using Smart Meter Data. In *2022 International Conference on Smart Energy Systems and Technologies (SEST)* (pp. 1-6). IEEE.

Chapter 2

Background and Literature Review

This chapter provides background information and reviews related work. Section 2.1 introduces load analysis, including load profiling and forecasting. Secondly, load analysis at different levels, i.e., the appliance level, the individual level, and the higher aggregation level, are discussed and reviewed in Section 2.2. Thirdly, enhancing techniques for load analysis, including transfer learning, online learning, and distributed and privacy-preserving machine learning, are introduced and reviewed in Section 2.3. The published work 1 and 3 also form part of this chapter.

2.1 Load Analysis

A general load analysis process is shown in Figure 2.1. The historical load data for load analysis can be divided into three categories, raw daily load data, averaged daily load data, and peak load. Raw daily load data are examined to investigate the day-to-day variations [169]. In contrast, averaged daily load data are used to determine typical intra-day load usage patterns of the residents. By analysing the peak, it is possible to identify the participants who are price-sensitive when determining the best electricity pricing strategy, and peak load analysis can be viewed as a particular branch of load analysis, which will not be discussed further here, and we refer to Dai et al. [39] for an in-depth analysis.

It is necessary to deal with missing values once smart meter data sets have been collected. Several commonly employed methods address missing values, including mean imputation, median imputation, multiple imputations, and hot deck imputation [176, 13].

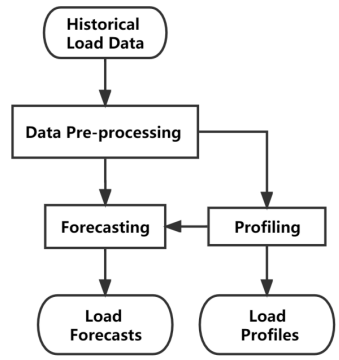


Figure 2.1: The general process of load analysis.

Further, input variables are often numerous when training a forecasting model, especially in the smart big data era. However, many variables may have unrelated characteristics with the target/ response variable, and variables may also be interdependent, easily leading to long training time and decreased forecast performance.

Feature transformation and feature selection are usually adopted to address the problem [60]. Feature selection is similar to feature transformation for both trying to reduce the number of input features. However, there are some differences between the two methods. Feature transformation aims to get transformed features by creating a new feature space, and the commonly used methods include principal component analysis (PCA), independent component analysis, and linear discriminant analysis. Feature selection [43] is to select a subset from the original feature space, and commonly used methods include filtering, wrapper, and embedding.

Moreover, the difference in magnitude among the variables in the dataset may cause the training algorithm to make inaccurate predictions. Besides, in real scenarios, load data often need to be normalized due to privacy requests [70]. Therefore, data normalization is a necessary preprocessing step for training the model. The commonly used data normalization methods for load analysis are zero mean normalization (Z-score normalization) [51] and Min-Max normalization [229, 182].

The load will be analysed after data preprocessing, with the profiling process being considered one of the key enhancing techniques for forecasting load. Electricity retailer and consumers can adjust their consumption patterns based on information smart meter data analysis. Nevertheless, due to the high variability of load patterns from different customers, the load analysis system may be computationally intensive and inaccurate. Profiling before the load forecasting can improve the forecasting

efficiency of the load analysis system. In real-world smart grid applications, load profiling is the most widely used and important session for enhancing load analysis. Therefore, this section will introduce load profiling and forecasting, and other enhancing techniques will be then discussed in subsection 2.3.

2.1.1 Load Profiling

Smart meter data profiling refers to classifying a large number of load curves or participants according to load usage behaviours [204]. For load profiling, clustering is the most commonly used method, and the general process of load clustering is shown in Figure 2.2.

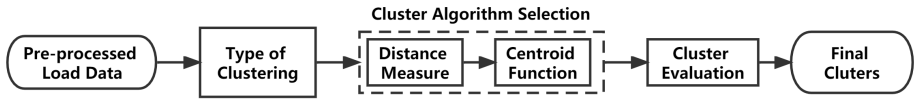


Figure 2.2: The general process of load clustering.

After pre-processing the historical load, one critical step is determining the cluster algorithms to cluster the load profiles. Hard and soft clustering are the two main types of clustering algorithms. In general, hard clustering involves assigning a load curve or set of load curves of a customer to a single cluster. In contrast, soft clustering involves assigning the load curves of individual customers to all the clusters based on probability [183]. The most common types of hard clustering are partitioning and hierarchical clustering, the latter including model-based and fuzzy clustering.

The partitioning clustering technique based on k-means, k-medians, and k-medoids is the most widely used in hard clustering, in which the cluster number needs to be determined. Yilmaz et al. [227] analysed the daily electric load patterns of 656 households in Switzerland and found that averaging the data suppresses the variety of household electricity usage patterns. Moreover, McLoughlin et al. [142] compared the clustering results of k-means, k-medoids, and self-organizing maps according to the Davies-Buldin validity index (DB-index). Customer profile classes were then built based on the selected algorithm. Finally, the paper combined the clustering results with questionnaire answers from the residents to summarize the unique characteristics of the electricity consumption of each PC, which well presented a straightforward way to connect the physical household information with the electricity consumption data.

Compared to partitioning clustering, hierarchical clustering offers greater flexibility and determinism. However, hierarchical clustering tends to be more compu-

tationally expensive. Using hierarchical clustering, Ozawa et al. [158] investigated the relationship between household lifestyles and electricity consumption. The experiment indicated that residents who follow a regular daily routine consumed less electricity than those who live a night-oriented lifestyle. Instead of focusing on the household level load clustering, Valdes et al. [197] employed hierarchical clustering for analysing the load profiles of three industrial plants in Chile, and the clustered profiles can then be used to optimize the electricity consumption management mechanisms.

The model-based clustering assumes that the load samples are derived from a mixture of components in a distribution [66]. Haben et al. [75] adopted the finite mixture model clustering to conduct an in-depth analysis of smart meter data to understand different electricity consumption behaviours better. In addition, bootstrapping was hired in this paper to verify the validity of clustering results. In Labeeuw et al. [112], household groups representative of Belgium were identified using expectation maximization clustering. The experiments discovered that the Belgian demand reduction potential is more significant during weekends and in the winter months by combining social information.

The concept of fuzzy clustering refers to the assignment of load curves to multiple clusters based on the degree of membership in each cluster. Jain et al. [97] proposed a cluster validation strategy based on fuzzy c-means clustering, which provides unbiased validity indices for electricity load profiles. Moreover, Viegas et al. [200] developed a clustering-based detection method for identifying non-technical electricity losses. In order to further improve the identification accuracy of the proposed method, the Gustafson-Kessel fuzzy clustering algorithm was employed, and the experiment demonstrated that the proposed method outperformed other state-of-the-art supervised and unsupervised learning algorithms, such as support vector machines and fuzzy c-means.

2.1.2 Load Forecasting

A well-developed load forecast framework can help the system operator optimize the demand side management strategy and help reduce greenhouse gas emissions and non-renewable fuel reliance. Moreover, the ultimate goal of demand management is to balance electricity supply and demand to maximize the benefits of the system. Table 2.1 lists the associated key stakeholders (grid operators, electricity retailers, electricity end-users, government) in the load forecasting framework and highlights the advantages of accurate load forecasting.

Load forecasting is often classified according to the forecast lead time, which can be roughly divided into long-term load forecasting, medium-term load forecasting,

Table 2.1: Associated key stakeholders and benefits of load forecasting

Electricity market stakeholders		Advantages of load forecasting
Grid operators		• Improve the utilization rate of power generation equipment • Reduce the cost of power generation and investment in power facilities • Alleviate the supply pressure of the grid
Electricity retailers		• Make reasonable tariff schemes so as to maximize profits • Offer energy-efficiency rebates to encourage customers to change load consumption behaviours
End-users	Commercial and industrial	• Improve the economic benefits and save production resources • Reducing environmental pollution through the dispersal of emissions
	Residential	• Save electricity bills and improve their living standard
Government		• Enable a reliable power supply system • Ensure economic growth and social welfare

and short-term load forecasting [106]. Long-term load forecasting generally refers to forecasting for 1 to 10 years. Medium-term forecasting refers to the forecast for a year. Long-term and medium-term load forecasting can generally be used to estimate the operating costs of power generation enterprises [106]. Short-term load forecasting refers to the prediction of daily load on an hourly or half-hourly basis, and the prediction results can be used for preventive control and emergency treatment.

The general process of load forecasting is shown in Figure 2.3.

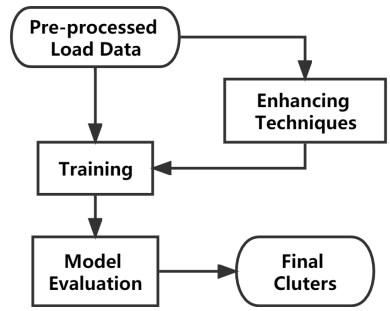


Figure 2.3: The general process of load forecasting.

Generally, two types of methods are commonly used for training load forecasting models, including statistical and machine learning methods.

The efficacy of machine learning methods, such as artificial neural networks (ANNs)[162] and support vector machines (SVMs) [215], have made them an attractive alternative to statistical methods such as the auto-regressive moving-average (ARMA), auto-regressive integrated moving average (ARIMA) [44], and Holt-Winters exponential smoothing [194].

Compared with machine learning techniques, traditional statistical forecasting models usually rely on statistical theory and historical data and often ignore the in-

herent characteristics of data, resulting in unsatisfactory prediction results. On the other hand, machine learning models have the characteristics of better performance, and satisfying training efficiency [10].

The high dimensional load data collected by smart meters make it challenging to train traditional machine learning algorithms. It has been recognized that load records can be seen as functional data, which has led to the development of a statistical field referred to as functional data analysis (FDA) [170] for the load analysis.

In the following subsections, we will first review statistical methods, including stochastic time series models and exponential smoothing, and then review popular machine learning methods, including SVMs, ensemble models, and DNNs. Moreover, the FDA will also be reviewed for load forecasting.

2.1.2.1 Statistical Models

For the statistical load forecasting models, the most commonly used methods are stochastic time series models [44], and exponential smoothing [194].

Stochastic time series models can be generally divided into: the auto-regression (AR(p)) model where p denotes the order of auto-regression; the moving average (MA(q)) model where q is the order of moving average; the auto-regression moving average (ARMA(p, q)) model; the auto-regression integrated moving average (ARIMA(p, d, q)) model where d denotes the order of integration; and the seasonal auto-regression integrated moving average (SARIMA$(p, d, q)(P, D, Q)_s$) model where P, D, Q are the seasonal parts of the model corresponding to p, d, q.

A SARIMA model may be written as [18]:

$$\phi_p(B)\psi_P(B^S)S_{y_t} = \theta_q(B)\tau_Q(B^S)\alpha_t \tag{2.1}$$

where: $S_{y_t} = \nabla^d \nabla_S^D y_t$, y_t is the load observed at time t. ∇^d and ∇_S^D denote the non-seasonal and seasonal difference (S is the seasonal length) operators, respectively, which transform y_t into stationary time series. B is the backshift operator, which is used to represent the backshift of time. When B is used for y_t, it means to reverse by one unit of time ($By_t = y_{t-1}$). For monthly data, $B_{12}y_t = y_{t-12}$ represents data from the same month of the last year. $\phi_p(B)$ and $\psi_P(B^S)$ are the non-seasonal and seasonal auto-regression operators, respectively. $\theta_q(B)$ and $\tau_Q(B^S)$ are the non-seasonal and seasonal moving average operators, respectively. p, P, q, Q denote the maximum backshift order for the non-seasonal, seasonal, auto-regression, and moving average operators, respectively. α_t is the white noise at time t. The above model can be represented by SARIMA$(p, d, q)(P, D, Q)_s$ where $s = 4$ represents the seasonal time series and $s = 12$ represents the monthly time series. When

$P = D = Q = 0$, the SARIMA model degenerates into an ARIMA model, and when $P = D = Q = p = d = q = 0$, the SARIMA model degenerates into the white noise process.

Exponential smoothing is a time series analysis method developed based on moving average (MA). Exponential smoothing predicts the load according to the weighted average of the historical time series. The recent data are given a larger weight whereas the previous data are given a smaller weight. This is based on the principle that the influence of a certain variable on subsequent behaviour is gradually attenuating [157].

A general exponential smoothing model for load forecasting can be written as [96]:

$$\hat{y}_{t+1} = \alpha y_t + \alpha(1-\alpha)y_{t-1} + \alpha(1-\alpha)^2 y_{t-2} + \cdots \tag{2.2}$$

where $\hat{y}_{t+1}$ is the forecasted load at time $t+1$, and $\alpha \in [0, 1]$ is the smoothing parameter that controls the weights decrease (exponentially).

Exponential smoothing can be divided into several different forms. Gardner Jr [68] provides a comprehensive review of exponential smoothing methods, in which 17 basic methods and some extensions based on these methods are described in detail. In general, single exponential smoothing is applied to sequences without trends or seasonality, and second exponential smoothing is applied to time series that only have trends. The triple exponential smoothing (also known as the Holt-Winters) targets sequences with trends and seasonality. When modelling the seasonal data, a Holt-Winters model consists of three equations, each with its smoothing parameters: trend, level, and seasonality components [212]. When the seasonal variations are constant and uncorrelated with time series, the additive Holt-Winters model can be hired. However, if the seasonal variables change proportionally with time series, the multiplication model can be chosen to predict the seasonal data.

2.1.2.2 Machine Learning-based Models

With the emergence of machine learning, machine learning-based load forecasting models have been adopted and gradually replaced statistical methods. ANNs are the earliest machine learning models for load forecasting. Based on ANNs, various advanced models such as ensemble models, SVMs, and deep neural networks (DNNs) have been developed and have become the hot spot in the load forecasting field.

ANN was proposed in 1991; it is inspired by the anatomy of the human brain and consists of artificial neurons in multi-layers for information communication. An example of the structure of an ANN is shown in Figure 2.4.

A typical ANN consists of the input layer, the hidden layer, and the output layer. Except for the input layer, each neuron in ANN is connected to neurons of

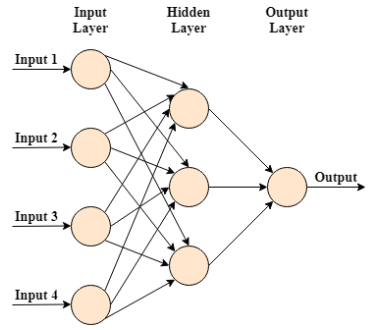


Figure 2.4: An example structure of ANN.

the former layer (i.e., the input neurons), with each connection corresponding to a weight. The sum of the product of all input and the corresponding connection weights are passed to an active function to calculate each neuron's final value, as is shown in Figure 2.5.

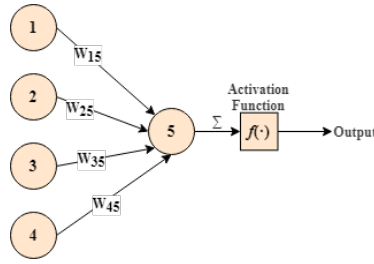


Figure 2.5: The calculation process of a neuron in ANN.

The activation function needs to be selected according to data characteristics, and the Sigmoid function is the most commonly used active function of ANN models [6]. One well-known ANN is the backpropagation (BP) neural network, a multi-layer neural network with error backward propagation. BP is widely used for its satisfying performance on prediction tasks. It, however, suffers from high computational cost; therefore, the radial basis function network (RBFN) was brought up to deal with this. The input variables of RBFN pass directly to the hidden layers without additional weights, and RBFN is proved to be less time-consuming than the traditional multi-layer neural network [69].

Ensemble learning trains multiple learners and aggregates each learner's predicted results to obtain the final output through combining strategies, which generally involve averaging, voting, and stacking [165]. According to the dependencies

between learners, one possible classification of the popular ensemble learning methods is as follows:

(1) Learners must be generated in sequence to satisfy the strong dependency between them (boosting).

(2) Learners are allowed to be simultaneously generated since there is no strong dependence between them (bagging and random forest).

Ensemble learning has been widely used in load forecasting in recent years. Ensemble models used in the reviewed studies are mainly: boosting, bagging, and random forest (RF).

Boosting adjusts the sample distribution according to the performance of the initial learner so that samples with the wrong prediction get more attention than others. Then it trains the next learner based on the adjusted sample. The process is iterated until a specified number of learner clusters are generated, or the aggregated learning criteria reach the stop threshold [165]. Commonly used boosting algorithms in the reviewed papers are adaptive boosting (AdaBoost), boosting trees, gradient boosting (GB), and extreme gradient boosting (XGBoost). Ahmad and Chen [9] adopted three machine learning models (ANN with nonlinear autoregressive exogenous multivariable inputs, multivariate linear regression, and AdaBoost) to predict load profiles one month, one season, and one year ahead at the district level. During training, datasets with different sizes were utilized for training models for different prediction intervals. This paper also adopted feature extraction to select essential variables, and the results showed that the AdaBoost outperformed other models significantly for all prediction intervals. Moreover, for seasonal forecasting, the error range of AdaBoost was relatively narrow, which indicated that the model trained based on AdaBoost was more capable of capturing the dynamic change of load curves. Zhang et al. [235] conducted short-term load forecasting for southern California. In this study, different models were adopted (multivariate linear regression, random forest, and GB), and the installed solar capacity was identified as an essential feature during the forecasting. The results of the comparative experiment revealed two insights: (1) The fact that the installed solar capacity became an important feature suggested that new and clean energy resources are important components in the system that researchers need to pay more attention to; (2) Different forecasting accuracy in different periods indicated that being able to capture the fluctuation of load curves is important for forecasting. Lu et al. [131] combined complete ensemble empirical mode decomposition with XGBoost to predict daily load consumption, peak load, and water delivery.

Bagging is based on bootstrapping sampling. It carries out multiple times of put-back sampling for a given dataset and trains learners simultaneously based on the obtained sampling set. When bagging is applied to a regression task, a simple mean or median can be adopted to obtain the final output [178]. de Oliveira and Cyrino Oliveira [45], for the first time, utilized bagging to forecast monthly load for countries with different development stages. The paper combined bagging with exponential smoothing and SARIMA and then used the simple mean and median to aggregate the results from single learners. A new variation of bagging, the remainder sieve bootstrap, was also proposed to enhance the forecasting results, and the result showed that the proposed method yielded the best MAPE for both developed and developing countries.

RF can be seen as an extension of bagging, which further introduces random selection in the construction of individual decision trees based on bagging.

The RF first uses bootstrapping to generate its training sets, and then a decision tree is constructed for each training set. Features are randomly selected, and optimization criteria are used to guide the splitting of nodes in constructing each decision tree learner. The prediction strategies of RF are: voting for the classification task and averaging for the regression task [165].

As the number of learners increases, RF generally converges to a smaller generalisation error than bagging. Moreover, the training efficiency of RF is often superior to bagging, benefiting from the randomness in constructing single learners. Wang et al. [206] adopted RF to predict hourly load usage patterns for two educational buildings in North Central Florida, and the feature importance distribution was also produced as a by-product. The proposed model was compared with the regression tree and SVM, and the results showed that the RF had the best superiority among all the trained models. Moreover, the feature importance distribution also proved that the influential features changed depending on different educational periods, which indicated that the load usage behaviour of educational buildings is highly related to different semesters.

As one popular machine learning method, SVMs can minimize actual risk by seeking risk minimization to get satisfactory forecasting performance. The variation of SVMs for regression problems is represented as support vector regression (SVR) [55], which is efficient for large-scale regression problems [35]. Similar to ANN, SVMs can cope with nonlinear and high dimensional data [51][232]. The disadvantage of SVMs is also similar to that of ANN for suffering from long training times with large data sets. Besides, the hyperparameters of SVMs need to be manually selected, which is also a complex step that needs to be optimized.

In Candanedo et al. [25], a comparative analysis for short-term load forecasting

for a building in Belgium was carried out, where RF was compared with multiple linear regression (MLR), SVR, and gradient boost machines. It demonstrated that RF has the best performance among the considered models. Based on actual historical data from the Tunisian Power Company, similar conclusions were reached in Lahouar and Slama [113] for day-ahead load forecasting. Furthermore, Li et al. [120] improved the short-term load forecasting accuracy by using a subsampled SVR ensemble (SSVRE) based on the SVR. The SSVRE is used to diversify individual SVR ensembles using a novel swarm optimization learning model, which has provided better forecasting performance and lower uncertainty for load forecasting.

DNNs are popular variations of NNs that have multiple hidden layers and have been proposed and applied to improve forecasting accuracy in recent years. For example, recurrent neural networks (RNNs) [199] are a type of DNN whose connections allow nodes to create a cycle, causing outputs from one node to influence inputs from another. Consequently, RNNs exhibit temporal dynamic behaviour and can process variable-length sequences of inputs using their internal memory. Zhang et al. [233] developed an RNN model to conduct short-term load forecasting. In particular, continuous and discrete time series are considered to generate multiple time series, which can then be modeled with an RNN to gain sequential information on the load. The proposed model outperformed the state-of-the-art models, such as the k-nearest neighbour and SVR.

Further, long short-term memory (LSTM) [84]is an advanced modified version of RNNs that keeps a constant flow of error between the nodes during backpropagation, which makes it capable of handling more complex problems than RNNs. In Kwon et al. [111], an LSTM model was proposed to conduct day-ahead load forecasting, which achieved high accuracy in forecasting the total load of the Korean power system in two years. Moreover, Muzaffar and Afshari [151] compared LSTM with statistical models, including ARMA, Seasonal ARIMA, and ARMA with exogenous inputs for load forecasting with various forecasting horizons from 24 hours to 30 days. Experimental results indicated that LSTM could increase forecasting accuracy by learning from long-term and short-term information in the load sequences.

2.1.2.3 Functional Data Analysis

FDA, as an important method to deal with high-dimensional big data has received increased attention in recent years. The FDA considers the discrete smart meter records as functional curves that can be seen as random functions in the realization of continuous stochastic processes. Therefore, applying FDA to load analysis benefits from the continuity of the functional curves, which account for the changing trend in data over a continuous period. Moreover, FDA can reduce the volatility

of smart meter data and achieve smoother load profiles by treating a set of data points as a single functional curve.

Many different approaches in the FDA have been developed, such as functional regression (FR), functional principal component analysis (FPCA), and functional discriminant analysis [171]. FR models are commonly used in FDA for load forecasting and can be classified into three types based on the predictors and response variables:

- Scalar predictors with functional response variables.
- Functional predictors with scalar response variables.
- Functional predictors with functional response variables.

The FR has been extensively studied in a variety of forecasting applications [188, 106, 65]. Shah and Lisi [187] compared the vector auto-regressive, functional auto-regressive, and functional non-parametric regression models for day-ahead load forecasting, and the results revealed that the functional non-parametric model outperformed the other two algorithms. Kiani and Zeng [105] used a functional time series regression approach to forecast the load of four regions. It turned out that the proposed method has better performance than ARIMA.

Further, Yao et al. [222] proposed an end-to-end functional deep neural network with a nested basis layer that was shown to be efficient at capturing functional data characteristics. Moreover, Perdices et al. [163] combined the FDA with the auto-encoder neural networks to characterize the network services, and the experiments demonstrated the complementary nature of the FDA and neural networks. Shah and Lisi [188] utilized functional auto-regressive models to forecast electricity prices. Kiani and Zeng [106] adopted functional B-spline approximation for the history load to explore the performance of models built on the individual area or aggregated data.

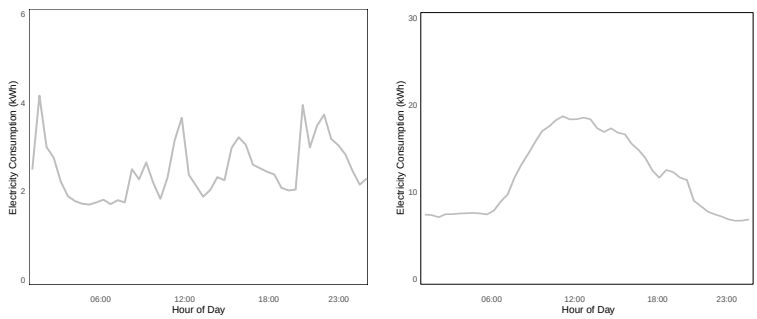
On the other hand, clustering functional curves prior to functional forecasting is a common way for the FDA to assist the functional forecasting models. For example, Chaouch [31] adopted hierarchical clustering to improve one-day-ahead non-parametric functional time series forecasting. Moreover, Martínez-Álvarez et al. [139] employed the funHDDC, a functional clustering algorithm, to assist the functional time series forecasting, which adopted a majority voting system to determine the best cluster number.

2.2 Multi-level Load Analysis with Smart Meter Data

According to forecasting space scales, the load analysis for the smart meter data can be divided into the appliance, individual building, and higher aggregation levels,

each serving a specific purpose. The appliance level load analysis allows consumers to identify their electricity consumption patterns for individual appliances, participate in demand side response through smart meters, and make informed decisions regarding their electric bills. The most popular method for analysing appliance level load is non-intrusive load monitoring (NILM) [81], in which the operational status (ON/OFF) or the electricity consumption of each appliance is determined using only the aggregated load from all the appliances.

Individual building level load forecasting becomes increasingly crucial with emerging applications in demand response, microgrids, and peer-to-peer energy trading. Nevertheless, individual buildings typically exhibit a high degree of randomness, thus making forecasting difficult. Alternatively, load forecasting at a higher aggregation level is typically used in community energy management and grid operations applications. Figure 2.6 gives an example of the daily smart meter readings at the individual building level and the aggregation level.



(a) Individual level. (b) Aggregation level.

Figure 2.6: Daily smart meter readings.

Simply aggregating the load consumption will ignore some essential information in individual buildings. It is, therefore, vital to select appropriate forecasting levels based on the type of application scenario. This section presents a review of studies on different levels of load analysis.

2.2.1 Appliance Level Load Forecasting

The appliance level smart meter data can be used for household energy system management by coordinating and optimizing appliance schedules based on individual preferences and electricity tariffs [225]. The most widely studied method for appliance level load analysis is NILM, which is based on machine learning and is

effective at disaggregating smart meter readings into appliance-level consumption.

NILM was first proposed by Hart [81] in the 1980s, and it aimed to decompose the electricity consumption of target appliances through non-intrusive detection. Compared to intrusive monitoring, NILM has lower deployment costs and higher user acceptance.

NILM is typically implemented for residential/household smart meter data. The Hidden Markov Model (HMM) is the most commonly utilized model. Kelly and Knottenbelt [102] developed a sparse HMM and achieved good load decomposition results. Both Makonin et al. [137] and Kong et al. [110] proposed improved methods based on HMM. Among them, Kim et al. [107] tried to introduce the time factor into HMM, but the proposed model has relatively high computational costs. Kong et al. [110] proposed a NILM model based on segmented integer quadratic constraint programming (SIQCP), which was proved to be efficient on load disaggregation. With the development of artificial intelligence and deep learning, various modern NILM models combined with artificial neural networks have been developed. Kelly and Knottenbelt [102] compared HMM with a convolutional neural network (CNN) for NILM, and the results showed that the CNN-based model outperformed HMM. In recent years, DNNs have provided new opportunities for the electrical utility industry [145], and are the most representative structures applied to NILM [102, 108], which have been proved to be more effective than other traditional statistic models.

Nevertheless, there are also studies focusing on conducting NILM for commercial and industrial smart meter data. For NILM to be applied in an industrial setting, new assumptions must be made due to differences in appliance states and temporal dependencies in industrial settings compared to residential settings [89]. Consequently, sub-metering has to be performed according to the size of the industrial building and the type of appliances used to disaggregate the industrial loads. Holmegaard and Kjærgaard [89] gives an analysis of how industrial equipment challenges NILM algorithms, including the fact that equipment events seldom change in load level, that equipment states are difficult to disaggregate, and that power change events of multiple types of equipment are often overlapping. To address these challenges, they investigated different levels of sub-metering to increase disaggregation accuracy and introduced a day-specific training based on factorial HMM (FHMM), which reduced the mean normalized error by half. Further, Martins et al. [140] compares the FHMM with a DNN-based model on a real-world site meter from a Brazilian factory. Results showed that the DNN-based model outperformed the FHMM model in normalized disaggregation and signal aggregated error for six industrial appliances.

2.2.2 Individual Level Load Forecasting

Recent years have seen an increase in the development of individual level load forecasting for dynamic demand modelling and real-time pricing systems. load forecasting at the individual level is a key session for distributed energy forecasts, but it is challenging due to the uncertainty associated with customer behaviour. The individual level load can be mainly divided into the residential/household load, and the commercial/industrial load [77]. The residential/household load refers to the electricity consumed by households where people regularly live, whereas commercial/industrial clients consume significantly more electricity than residential customers.

Forecasting residential/household load can be helpful in energy consumption reduction and home energy system management. Forecasting residential/household load is essential for utilities in controlling residential demand and optimizing dynamic pricing strategies. Moreover, residential/household load analysis can facilitate the development of efficient incentive schemes to enhance control mechanisms through two-way communication. To forecast residential/household load, many statistical and machine learning models have been applied. To predict the consumption of four multi-story residential buildings in Seoul, South Korea, Ullah et al. [196] adopts a statistical approach by employing HMM. A sequence of floor occupancy values is created by transforming the load data into floor occupancy values based on HMM. With the wide application of the machine learning models for residential/household load forecasting, it has been demonstrated that the machine learning models are superior at extracting the nonlinear relationship between the load sequence and the associated stochastic factors [24] than the statistical models. In Cheng et al. [33], a convolutional neural network with squeeze-and-excitation modules and micro-meteorological data was used to develop a day-ahead probabilistic residential load forecasting model. This model incorporates feature extraction to assist in forecasting, and has been evaluated using real-world data from eight residential communities. Moreover, Zang et al. [231] combined LSTM with self-attention mechanisms to propose a hybrid model with two input channels, and experiments on the practical residential dataset verified its superiority. In Ryu et al. [181], two types of deep neural networks were used to perform short-term load forecasting. The results showed that the deep belief model had the best performance in terms of training accuracy and time consumption.

On the other hand, load forecasting can be complex in commercial or industrial buildings, particularly those with varying heating, cooling, and lighting requirements [226]. Furthermore, unlike residential/household buildings with regular load usage patterns, commercial/industrial buildings are affected by occupancy levels

and scheduling. Commercial building electricity loads have been forecasted using various techniques, including statistical models and machine learning models. Stochastic time series models are commonly used for commercial/industrial load forecasting. Nepal et al. [153] combines ARIMA with k-means clustering to forecast peak load for university buildings, which improves the forecasting accuracy of a single ARIMA model.

2.2.3 Higher Aggregation Level Load Forecasting

Power utilities rely heavily on load forecasts at the higher aggregation level to plan their day-to-day operations, scheduling, and load-shedding strategies. The aim of load forecasting at higher aggregation levels is to estimate a load for an entire region, city, province, or even country rather than dividing the load into small units. Aggregated load data can be collected directly from a substation or aggregated individual load data.

Similar to commercial/industrial load forecasting, stochastic time series models are commonly used for higher aggregation level load forecasting. Espinoza et al. [61] adopted periodic autoregression to extract the stationary properties of 245 load profiles from a Belgian grid, which are then used for load profile clustering. Moreover, Amjady [12] combined the ARIMA model with the knowledge of experienced human operators for short-term load forecasting. Based on a real-world dataset from Iran's power network, the proposed methodology was found to be more effective than the original ARIMA model without expert knowledge.

Recently, DNNs have become the most popular topic for forecasting load at higher aggregation levels. Barman et al. [16] presented a regional hybrid short-term load forecasting model that considered regional climate conditions and used grasshopper optimization algorithms. Compared with traditional hybrid models, the proposed model proved to have better accuracy. In Lv et al. [134], a hybrid model was developed based on variational mode decomposition and LSTM and evaluated on data from Singapore and the United States. The results indicated that the proposed model could achieve accurate forecasting of the power grid load.

However, unlike analysing load at the individual level, which is aimed at anticipating household load use patterns, forecasting demand at the higher aggregate level can be beneficial for estimating the generation capacity, making long-term investments, optimizing electricity pricing strategies, and expanding transmission capacity [91]. López et al. [130] proposed an online short-term load forecasting system to forecast the Spanish inland and the regional load one hour ahead. The proposed system was tested on two years of data. It was demonstrated that the system achieved satisfactory performance for different days by incorporating spe-

cial days such as holidays and weekends. Moreover, Liu et al. [125] conducted day-ahead hourly load forecasting at the province level. A greedy forward feature selection process was utilized to assist forecasting, and the comparative study revealed the classic linear models and their variations, i.e., the MLR, the multivariate adaptive regression splines, and the SVR, performed the best compared to other candidates, including ARIMA and two DNNs.

2.3 Enhancing Techniques for Load Analysis

Smart energy meters can provide real-time, high-frequency data to facilitate demand-side management and response and enhance economic and social well-being. Smart meter-based load consumption records are a common type of time-series data that offer detailed information about electricity usage in almost real-time. These records are known as smart meter big data, which pose challenges to traditional profiling and forecasting proceedures due to their large size and high dimensionality [7]. Therefore, this section reviews the related enhancing techniques for load analysis.

Typically, appliance level load analysis involves the collection of load consumption for each appliance, which is prohibitively expensive since smart devices (such as smart plugs) must be deployed in the living environment. As an advanced machine learning method, transfer learning (TL) can address the issue of insufficient labelled data by leveraging multiple sources of information. Therefore, we will first review TL as one of the enhancement techniques for load analysis.

Secondly, as real-time smart meter readings become available, online learning (OL) is necessary to leverage dynamic changes in consumption patterns, which enables online models to dynamically adapt to new patterns in load data without retraining the complete training dataset. Therefore, OL will be reviewed in this section as one of the enhancing techniques for load analysis.

Moreover, the readings of the loads are generally distributed among a large number of smart meters, which results in a high volume of network traffic when the data is transferred to a central server [192]. Additionally, training a centralized model for all smart meter users requires data sharing, posing security and privacy concerns, and is challenging to comply with stringent privacy regulations [195]. Therefore, advanced distributed and privacy-preserving machine learning techniques will also be reviewed in this section.

2.3.1 Transfer Learning

Most of the traditional machine learning algorithms assume that the training and test data have similar distributions and feature spaces. However, this assumption

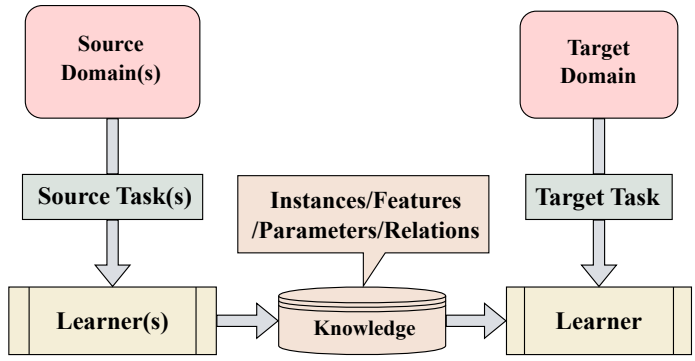


Figure 2.7: Transfer learning process.

does not hold in the majority of real-world scenarios. Furthermore, traditional machine learning has been hampered by a lack of adequately labelled training data and mismatched computing capability. TL [159] was proposed to address these challenges by leveraging knowledge from a single or multiple source domains to enhance a training task in the target domain (Fig. 2.7). The knowledge transferred could be instances from source domains [210], shared features from source domains and the target domain [129, 177], parameters from the trained learners of source domains [214], or relations between source domains and the target domain [193].

According to different implementation scenarios, TL can be categorized as single source TL and multiple sources TL. Single source TL refers to transferring knowledge from a single source domain [190] whereas the multiple sources TL utilizes several source domains to transfer the knowledge [221, 67]. Moreover, different TL techniques have been proposed to handle similar or different data structures between the source and target domains, i.e., homogeneous and heterogeneous TL [83, 241].

According to different label settings, various TL methods have been proposed and can be classified into three major categories, i.e., transductive, inductive, and unsupervised TL [159].

Inductive TL is used when the target domain has well-labelled data, and there are different tasks in the source and target domains. TrAdaBoost [41] is a well-known inductive TL technique that extracts valuable information from the source domain by re-weighting predicted instances in both the source and target domains. However, this method only utilized a single source domain, and the extracted information may not be sufficient for the training task in the target domain. To address this challenge, Li et al. [121] and Ye et al. [223] combined the transfer task with

multiple source domains, which enhanced the training performance of the target model. Moreover, unlike Dai et al. [41], which retained only one base learner and discarded the rest, Eaton and desJardins [58] assumed that all base learners are useful, based on the theory that older learners can represent the major distributions of instances, while newer learners can provide accurate information about subsequent iterations.

Transductive TL is used when the source domain data is labelled, but the target domain data is unlabelled, and both the source and target domains have the same task. Domain adaptation is the most well-known subfield of transductive TL [154], which aims to minimize the marginal distribution gap between the source and the target domains. Xia et al. [209] proposed a method for selecting and weighting instances based on PU learning, a set of semi-supervised methods used to train a binary classifier, to identify examples from the source domain that are most likely to improve the training task. However, this method was limited by the difficulty of dealing with high-dimensional distributions. A solution was provided by Xu et al. [210], using the logistic approximation to adapt the high-dimensional data from the source domain to the target domain.

In real-world situations, both the source and target domains may lack sufficient well-labelled data, which cannot be addressed by the TL techniques discussed so far. As a solution, unsupervised TL was introduced. Wang et al. [207] proposed transferred discriminative analysis, a method for generating class labels for unlabelled target data by leveraging knowledge from the source domain. Although unsupervised learning is a more practical solution in TL, it has received little attention from researchers over the last decade.

2.3.2 Online Learning

Online learning is a machine learning paradigm for real-time data that uses feedback from sequence data to learn and update the best predictor for future data. Compared to the optimal model in foresight, the primary goal of OL is to minimize cumulative error across the whole data sequence [87]. OL is generally more effective and scalable when dealing with large-scale real-world machine learning problems involving data of varying quantity and velocity than conventional batch learning algorithms, which require pre-given training data.

OL has been extensively investigated for many years [29, 86]. There are two fundamental types of OL algorithms: first-order OL and second-order OL [86]. The Perceptron [174, 155] is one of the earliest first-order OL algorithms, relying on gradient feedback to update a linear classifier whenever a new sample is misclassified. Passive-Aggressive (PA) [37] was introduced as a family of first-order OL

algorithms based on margin-based learning. It updates the model when the classification confidence of a new sample falls below a predefined threshold. Moreover, online gradient descent [244, 17, 48] was proposed to model the OL as an online convex optimization problem.

The misclassified instances are retained as support vectors (SVs) in standard OL algorithms (e.g., Perceptron and PA). Despite their solid theoretical guarantees and efficient functioning, a fundamental issue is that the increasing number of SVs over time may result in an increased computational overhead. To overcome this challenge, Dekel et al. [47] discarded the oldest SVs, assuming they were less representative of the data streams. Additionally, Zhao et al. [238] presented bounded online gradient descent to constrain the number of SVs that fall below a threshold.

Unlike first-order OL algorithms, which maximize convergence by utilising only the first-order derivative/gradient information of the cost function, second-order OL algorithms maximize convergence by utilising both the first-order and second-order information. The second-order Perceptron algorithm [30] was designed to examine the geometric properties of data. In order to capture second-order information about the confidence level of the features, the confidence-weighted algorithm [54] was developed to manage the updating of the classifier. Furthermore, the second-order OL requires exponential space and time for updates, and the sketched online Newton [132] was introduced to address this issue. The sketched online Newton is an enhanced version of the online Newton step with a linear running time in dimension and sketch size, allowing for dramatic improvements in second-order learning efficiency.

2.3.3 Distributed and Privacy-preserving Machine Learning

The goal of distributed machine learning is to distribute tasks with large volumes of data and computation to multiple machines to improve the computational efficiency and scalability. With the rapid exponential growth of smart meter data, machine learning frameworks increasingly need the distributed support of machine learning algorithms.

Liu et al. [124] elaborated on the development trend of distributed machine learning and proposed that the performances of the machine learning models need to be improved by enhancing algorithm design and optimization methods. Zhou et al. [242] designed a kunpeng distributed computing platform, which combined the large-scale distributed system with the optimization algorithm. This platform wraps complex communication and scheduling into the application program interface to quickly realize model synchronization. It supports directed acyclic graphs and various data synchronization and has strong fault tolerance. Nazari et al. [152]

used an advanced coding method to solve the time-consuming problem of data communication and matrix calculation, which significantly improved the model training efficiency.

When data owners want to combine their local data with training a deep neural network model, the traditional way is to integrate their data into the central server and use the integrated data to train a standard centralized model. However, data uploading and integration often involve legal issues such as privacy leakage and stolen data, making the centralized model training scheme more complicated. Many studies designed privacy protection schemes for load analysis, which can be mainly divided into identity privacy and data privacy protection.

Identity privacy protection refers to implementing user identity recognition while hiding the real ID of the users by unique mechanisms such as blind signature [59]. For instance, Cheung et al. [34] adopted a blind signature to protect the identity privacy of smart meter users. In the proposed scheme, the power company has access to real-time electricity data but does not know the identity of owners, so the privacy of users is protected.

On the other hand, data privacy protection is usually achieved by adding noises or encryptions to electricity consumption data. Kalogridis et al. [100] suggested that the load signatures of appliances in individual households could be moderated by home electrical power routing. Gündüz et al. [74] studied the sensitive data in smart metering from an information-theoretic perspective. In the paper, the smart meter readings were diversified by the alternative energy source, and the storage units filtered the real consumption data. Cao et al. [26] proposed a fog computing approach based on differential privacy against NILM, which adds noises to the behaviour parameter derived from the FHMM rather than the original consumption data. The proposed scheme achieved a satisfying trade-off between data utility and privacy. Hassan et al. [82] compared four variants of differential privacy in blockchain-based smart metering, and the experiment showed that such mechanisms could provide an effective privacy-preserving scheme.

Moreover, federated learning (FL), which performs distributed joint modelling while protecting the privacy of the original data, has the potential to be applied to load analysis. FL has been proposed for training a global model from data distributed across multiple devices with only intermediate updates periodically being sent to a central server [217], which can be categorized into horizontal FL, vertical FL, and federated transfer learning (FTL), depending on how data are distributed among different devices in the sample and the feature space.

Horizontal federated learning (HFL) refers to the situation in which data from distributed devices share the same feature space but differ in samples. Vertical

federated learning (VFL) was proposed on the premise that heterogeneous data from various devices share common sample IDs but have distinct feature spaces. Thus VFL focuses on the correlation between devices from different sectors. In a typical VFL process, data with common sample IDs are retrieved and used to train a machine learning model. Distinguished from HFL and VFL, FTL [126] refers to situations where data across multiple devices differ in terms of both feature spaces and sample IDs and is regarded as a significant extension of traditional FL frameworks [217]. By enabling users to leverage large datasets with well-trained machine learning model parameters, FTL goes beyond simply allowing users to exploit only matching data (i.e., data with overlapped feature spaces or sample IDs) [149].

Google pioneered HFL by utilising data distributed across many local Android devices to forecast text input without violating privacy regulations [144]. Abad et al. [3] then developed a hierarchical heterogeneous HFL architecture for extending HFL to heterogeneous environments, thus optimizing the communication efficiency in local source devices with heterogeneous networks. Additionally, Bonawitz et al. [21] designed a secure aggregation scheme based on McMahan et al. [144] to further enhance the privacy of aggregated intermediate updates. Further research [185, 201] has been proposed to address the high cost of communication in the HFL framework.

VFL is more difficult to implement than HFL since it requires encrypted user-ID alignment algorithms [218] for common entities [217] and the authentication of a fully trusted third-party. To overcome these obstacles, Yang et al. [219] developed a framework that eliminates the need for a third-party coordinator, and this framework has proven to be efficient and scalable. Although VFL can handle heterogeneous domains, most VFL techniques rely on statistical models such as logistic regression rather than sophisticated machine learning frameworks, indicating that this field still demands enormous effort.

FTL has received growing interest in real-world applications, such as smart healthcare [32], traffic monitoring [136], smart energy [236], and image analysis [216]. The majority of current FTL systems are based on deep learning architectures [32, 236, 216, 104] that usually freeze the base layers of the global model and retrain the fully-connected layer on local devices. Chen et al. [32] performed human activity recognition via FTL, which replaced one of the fully-connected layers with a correlation alignment layer to facilitate domain adaptation. FTL with deep learning architectures is efficient due to the highly transferable features in the low-level layers and the ability to capture specific features in the high-level layers of the deep network [78].

Since the model parameters still need to be transferred, the FL provides lim-

ited privacy guarantees for the smart meter data. A prior study has proposed differential private FL (DPFL) to provide clients with stronger privacy guarantees, which has been used as the basis for many privacy-preserving FL-based schemes [27, 95, 166, 202]. Privacy in FL can be divided into global differential privacy FL (GDPFL) [208], and local differential privacy FL (LDPFL) [27] based on different noise-adding mechanisms. In GDPFL, the trusted server applies the noise during the parameter aggregation, whereas in LDPFL, each participant adds noise to the model parameters before uploading them to the server.

2.3.4 Discussion

TL, FL, and OL are all innovative approaches built on standard machine learning techniques and has been applied in many popular fields. In this subsection, we will discuss their implementation scenarios to investigate the most appropriate technique for each load analysis level. Table 2.2 compares the implementation scenarios of traditional machine learning, TL, FL, and OL.

Table 2.2: Implementation scenarios of enhancing techniques

	Decentralization	Heterogeneity	Inadequate Well-labelled Data	Privacy-Preserving	Client-Side Personalisation	Real-time Big Data
Traditional Machine Learning	×	×	×	×	×	×
Transfer Learning	×	✓	✓	×	✓	×
Federated Learning	✓	✓	✓	✓	✓	×
Online Learning	×	×	×	×	×	✓

Traditional machine learning relies on a massive amount of well-labelled centralized data and assumes that all data collected are homogeneous [154] to build the load analysis models. However, many real-world load analysis tasks require a more scalable, private, and dynamic machine learning framework that can manage real-time load sequences from IoT devices. TL, FL, and OL were therefore proposed as enhancing solutions to build advanced models.

TL enhances target model performance by providing learners in target domains with a baseline performance rather than starting from scratch, thereby reducing computational overhead [122]. The appliance level load analysis requires sufficient labeled dataset from individual appliances to accurately extract the load patterns from historical aggregated load, Therefore, combining TL with appliance level load analysis is a suitable choice, which will be discussed in Chapter 3.

In the new era of big data, a prominent application scenarios is modelling real-time smart meter data, which typically become obsolete within hours or even min-

utes [239], such as real-time non-intrusive load monitoring systems for elderly living alone [11]. Additionally, there is a *cold start* [118] problem in real-world load analysis tasks, which refers to new clients or datasets incoming into the load analysis system at the individual building and higher aggregation levels. Thus, it is vital to incorporate the OL paradigm with load analysis models to overcome these constraints, which will be explored in Chapter 4.

FL is applicable in situations where the datasets are heterogeneity between individual client devices. Besides, FL keeps the raw data stay in the local client side and only requires the updating of the local model parameters, which provide a certain level of privacy guarantee for the model training process. The appliance level load analysis has the potential to reflect the daily routines of the clients, and thus may raise privacy concerns. Moreover, different clients may have different appliance usage habits, which produce heterogeneous load usage patterns. Therefore, applying FL to appliance level load analysis is a more practical way to train a well-performed central model without exposing the privacy of the clients, which will be further discussed in Chapter 5.

2.4 Chapter Summary

This chapter introduces the background of smart meter-based load analysis and reviews the related state-of-the-art. The two essential steps of load analysis, load profiling and load forecasting are reviewed, followed by a multi-level load analysis discussion. Furthermore, enhancement techniques are presented for enabling smart meter data analysis in distributed and real-time settings, which is necessary for understanding the remainder of this thesis.

Nevertheless, most research has focused on building precise models with minimal errors using an pre-given/offline dataset in a centralized environment. However, few studies have explored the effectiveness of applying the above mentioned enhancement techniques to load demand analysis at different levels. On the other hand, the reviewed enhancement techniques has been proved in other fields, but they are rarely used for load demand analysis. Therefore, this thesis explores different enhance techniques to analyse different levels of load consumption.

Chapter 3

Load Profiling and Forecasting System for Multi-Level Smart Meter Data Analysis

3.1 Introduction

The installation of smart meters has made it possible to access high-resolution load consumption data, which has increased the accuracy of load forecasting at the aggregation level and enabled data-driven load forecasting at an individual building level and even appliance level. This chapter aims to analyze smart meter data at different levels in a traditional offline setting.

NILM using smart meter readings is an ambient intelligence solution for many modern application scenarios. For example, a NILM framework combined with abnormal detection techniques can provide telehealth care for elders and patients with care needs, reducing burdens on caregivers and the health system. In the first part of this chapter, in order to analyze appliance level smart meter data, a novel deep neural network model for NILM has been proposed, which disaggregates household electricity usage into individual appliance consumption utilising the sequence-to-point model and transfer learning. The daily behaviour regularities of residents are then inferred by combining principal component analysis and k-means clustering based on the disaggregated appliance level consumption. The appliance level load forecasting framework of this part is adapted from our paper 6.

Rather than extracting the load patterns of individual appliances from historical aggregated load, smart meter data analysis at building and higher aggregation levels make predictions of future load patterns. It is essential to develop customized

and effective load forecasting frameworks at the individual building level to accommodate the differences in load consumption behaviour within each building. However, individual buildings usually exhibit high randomness, the smart meter data collected from which may hinder the forecasting performance of the models, making the forecasting problem at the building level more challenging. Meanwhile, aggregating smart meter readings across buildings can reduce randomness and improve forecasting results. Therefore, forecasting models may perform differently at different aggregation levels because the predictability of smart meter data varies.

3.2 Problem Statement

As aforementioned, the load analysis for the smart meter data can be divided into the appliance, individual building, and higher aggregation levels. The appliance level load analysis allows consumers to identify their electricity consumption patterns for individual appliances, participate in demand side response through smart meters, and make informed decisions regarding their electric bills. The most popular method for forecasting appliance level load is NILM [81], in which the operational status (ON/OFF) or the electricity consumption of each appliance is determined using only the aggregated load from all the appliances. Table 3.1 presents the key challenges discussed in this chapter, along with their corresponding applications and benefits for different load analysis levels.

Individual building level load forecasting becomes increasingly important with emerging applications in demand response, microgrids, and peer-to-peer energy trading. Despite the challenges of predicting individual building energy usage due to its randomness, higher-level load forecasting is often utilized in community energy management and grid operations applications.

The last few decades have seen the development of many smart technologies to assist people in their daily lives [115] [135], such as intrusive load monitoring to help residents better understand their appliance usage patterns. The majority of these devices, however, are required to be intrusively attached to users or installed in residential environments, which will have an unpredictable negative psychological effect since many residents tend to reject noticeable devices [228]. As a result, delivering various smart services utilising assistive technology that emphasizes self-management and autonomy is more desirable.

NILM using smart meter readings, is an ambient intelligence solution for appliance level smart meter analysis, which utilizes algorithms to derive the state and load consumption of the individual appliance in the households based on their smart meter readings. The disaggregated results can then be used for further analysis to

Table 3.1: Summary of focused challenges, applications and advantages of different load analysis levels

Load Analysis Levels	**Focused Challenges**	**Applications**	**Advantages**
Appliance Level (NILM)	Inadequate well-labelled data, client-side personalization	Demand response, microgrids, and peer-to-peer energy trading	Delivers various smart services utilising assistive technologies
Individual Building Level	High-resolution smart meter data analysis, seasonal load usage pattern variations, model performance	Energy system operation, peer-to-peer energy trading	Essential for the reliable and economical grid operation
Higher Aggregation Level	Regional load consumtion analysis, model performance	Distribution network operations	Important to the planning of building energy systems

monitor the activities of daily life (ADLs) of the smart meter users and timely detect abnormal behaviours combined with the abnormal detection techniques. Compared to intrusive monitoring, which focuses on hardware deployment, non-intrusive monitoring hardware facilities are simple and easy to install and maintain. Moreover, smart meter big data are considered a cost-effective source to achieve NILM with the rolling out of smart meters [5]. Using electricity consumption data (e.g., current, voltage, active power) collected by smart meters to infer ADLs [101] is a practical and commercially feasible method. Therefore, this section proposes an ADLs pattern recognition NILM framework to analyze appliance-level smart meter data.

Buildings account for up to 40% of the global energy consumption [164] as an indispensable role in day-to-day life. Building load forecasting is vital in improving the flexibility and reliability of energy system operations [146]. With the installation of smart meters, building-level high-resolution load consumption data become available, providing more opportunities for load forecasting applications at the building level (e.g., peer-to-peer energy trading) and aggregation level (e.g., distribution network operations).

Building load forecasting is essential for the reliable and economical operation and planning of building energy systems. Building load forecasting models can generally be categorized into the building physics-based approach and the data-driven approach. The former requires detailed physical information (such as building material types and ventilation system parameters which are often difficult to obtain) to calculate the thermal dynamics and load consumption behaviours of individual buildings [205]. Moreover, the physics-based approach often leads to unsatisfactory forecasting results due to inevitable errors in the process of information collection [180]. In contrast, the data-driven approach, such as machine learning-based forecast models, which uses real-world data to find the hidden relationship between independent variables and the response variable, has been widely emphasized in building load forecasting during the past two decades for its superior performance [213].

Traditionally, building load forecasting is conducted to forecast at the aggregation level, such as the hourly aggregation readings in a community. In contrast, high-resolution data are rarely available for load forecasting at individual buildings. The widespread deployment of smart meters makes it possible to forecast and analyze building level load consumption with greater precision [147]. Individual meter readings can be collected by smart meters so that different levels of aggregation can be analysed. Data collected from smart meters for individual buildings indicate that the load patterns of these buildings are largely determined by the behaviour of the building residents, which is highly unpredictable, as discussed in Section 2.2. Consequently, load forecasting models perform worse on individual level smart meter data than aggregated data at a higher level. To improve forecasting accuracy for individual level load forecasting, rather than simply applying advanced forecasting models, it is necessary to use enhancing techniques such as feature engineering during the data processing stage.

Motivated by the above analysis, we propose a consensus-based load profiling and forecasting system, which firstly identifies the underlying physical factors that affect the electricity consumption behaviours in different seasons and track the dynamic cluster trajectories of the 169 real-world buildings throughout the whole year. In addition, key driving factors influencing electricity consumption patterns are identified by quantifying the importance of the load-related features. Different forecast models are explored at both the building and aggregation levels.

3.3 Appliance Level load Analysis: An ADLs Patterns Recognition NILM Model

3.3.1 Preliminaries

We introduce several essential concepts related to the proposed ADLs Patterns Recognition NILM Model in this section.

3.3.1.1 Non-intrusive load monitoring

Given the aggregated load L_t at time t:

$$L_t = \sum_{i=1}^{I} l_t^i + \gamma_t, \tag{3.1}$$

the goal of NILM is to recover the status of I target electrical appliances. l_t^i and γ_t denote the load consumption for the i-th appliance and the residual/unmonitored load respectively at time t. NILM can be formulated as either a classification or a regression task depending on the status variables of individual electrical appliances we aim to recover.

For the regression task, the NILM model aims to find the approximation, denoted as F, of the true relationship between the aggregated household-level consumption (L_t) and the appliance-level consumption

$$\boldsymbol{L} = [\hat{l}_t^1, \hat{l}_t^2, \ldots, \hat{l}_t^i, \ldots, \hat{l}_t^I] = F(L_t), \tag{3.2}$$

where $\boldsymbol{L}$ is the predicted load consumption sequence of I target electrical appliances at time t.

For the classification task, thresholds need to be set for the NILM model to determine the states (e.g., ON/OFF) of each target appliance. A commonly used threshold method is the activation-time thresholding, which could avoid the negative effect of the abnormal spikes during the OFF state to improve the inference accuracy [102]. For the sake of simplicity, we assume that there are two typical states (ON/OFF) for the target appliances, and the state s_t^i for i-th appliance at time t is related to its threshold λ^i

$$s_t^i = \begin{cases} 1, & l_t^i \geq \lambda^i \\ 0, & l_t^i < \lambda^i, \end{cases} \tag{3.3}$$

where 0 represents the OFF state, and 1 denotes the ON state. Therefore the classification task for NILM can be defined as

$$\boldsymbol{S} = [\hat{s}_t^1, \hat{s}_t^2, \ldots, \hat{s}_t^i, \ldots, \hat{s}_t^I] = F_s(L_t), \tag{3.4}$$

where $\hat{s}_t^i$ is a binary variable indicating the predicted ON/OFF state of i-th electrical appliance at time t.

In the real scenario, the general goal of NILM is to infer the electricity user behaviour of each household from their smart meter readings. NILM can be formulated as either a classification task or a regression task depending on the status variables of individual electrical appliances we aim to recover.

3.3.2 Framework Design Overview

By utilising NILM, the proposed ADLs pattern recognition NILM framework is intended to infer the ADLs of smart meter users to assist in various real-world application scenarios, including monitoring patients in real-time to ensure consistent tracking of disease deterioration and implementation of timely intervention measures. Fig. 3.1 shows the proposed framework.

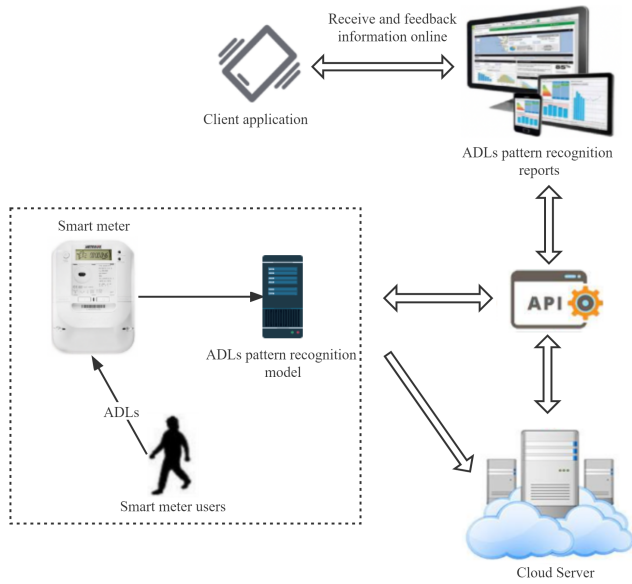


Figure 3.1: The proposed ADLs pattern recognition NILM framework.

The aggregated load consumption of the users is summarized by the smart meter and automatically upload to the ADLs pattern recognition model. After learning

valuable signatures for each appliance, the model can disaggregate the target appliances' consumption related to ADLs and then summarize the ADLs pattern of the smart meter users. Then, ADLs pattern recognition reports will be generated and delivered through the client application to the related users

Since accurate load disaggregation remains a recognized challenge, we propose an ADLs pattern recognition model based on an improved NILM architecture. The disaggregated appliance level loads are then further analysed using PCA and k-means to detect and visualize the ADLs of the smart meter users. The use of PCA aims to extract more orthogonal features from the disaggregated results. Then, we choose to use k-means clustering to classify the extracted features from PCA since it is easy to implement. The following sections describe the details of ADLs pattern recognition and related methods.

3.3.3 Implementation

The ADLs pattern recognition model implementation contains two parts, as shown in Fig. 3.2. The first part (green) is an improved NILM architecture based on sequence-to-point and transfer learning. In this part, valuable signatures for representative appliances will be learned and the target appliances' consumption related to ADLs will be disaggregated. The second part (blue) applies PCA and k-means to detect and visualize the ADLs of the smart meter users based on the disaggregation results.

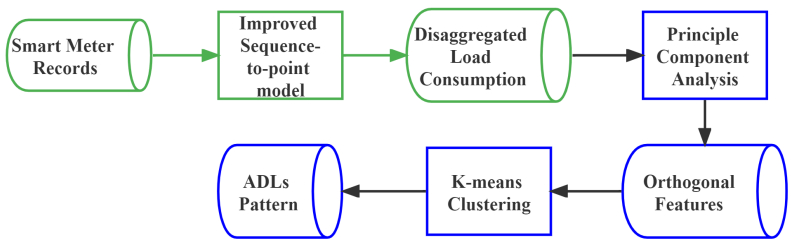


Figure 3.2: ADLs pattern recognition model implementation.

3.3.3.1 An improved NILM architecture

Electrical devices are divided into switch type, finite-state type, and continuous variable type [81]. The switching devices only have one status after starting, such as kettles. Finite-state devices allow transitions between multiple states once turned

on, such as washing machines and dishwashers. Continuous variable devices have infinite states, such as sewing machines. Since the energy consumption of the continuous variable devices is usually insignificant, it is not considered in our framework. The earliest NILM algorithm was based on combinatorial optimization, aiming to determine the best combination state of the monitored electrical appliances and make the total combined power as close as possible to the real meter reading. The DNN was firstly applied to NILM by Kelly and Knottenbelt [102] where experiment results indicated its superior performance over traditional methods such as the FHMM.

Unlike the traditional deep neural network, the sequence-to-sequence model uses a sliding window to get the neural network input. By dynamically setting various possible sliding window sizes, the output value of the sliding window at each time step is predicted and averaged. This method has been proven effective in NILM [102]. However, in the sequence-to-sequence model, the element of the output signal needs to be computed multiple times, which brings redundancy to the calculation. The sequence-to-point model [234] is brought forward to solve this problem. The input of the sequence-to-point model is also a sliding window applied to the main power supply, and the output is a single point from the target appliance. The sliding window ensures that the model learns contextual information of the data, and the output is only the midpoint of the window, which can significantly reduce the computational cost during training.

The original model in D'Incecco et al. [52] can achieve satisfactory disaggregation accuracy, but it has the same limitations as other deep neural network models, such as long training time and high computational cost. To deal with this, an improved NILM architecture based on the above model was proposed, as shown in Fig. 3.3. Specifically, dropout with a rate of 0.5 was used in CNN to improve the generalisation ability of the model. Moreover, one-dimensional CNN is used to replace two-dimensional CNN. Besides, we use the trained convolution neural network parameters of the washing machine as the pre-trained model for other appliances and then only train the dense layer for other appliances to complete transfer learning. In the proposed model, one-dimensional CNN was adopted to pre-train the model based on the washing machine, which was proved to be able to learn more active channels than other appliances (kettle, fridge, microwave, dishwasher). The individual consumption patterns of other appliances were then fed into the dense layer and combined with the pre-trained CNN (frozen part in Fig. 3.3) to update the final model to achieve appliance transfer learning. Our improved neural network model is expected to accurately conduct NILM for the target appliances from the unseen house and improve the training efficiency of the model.

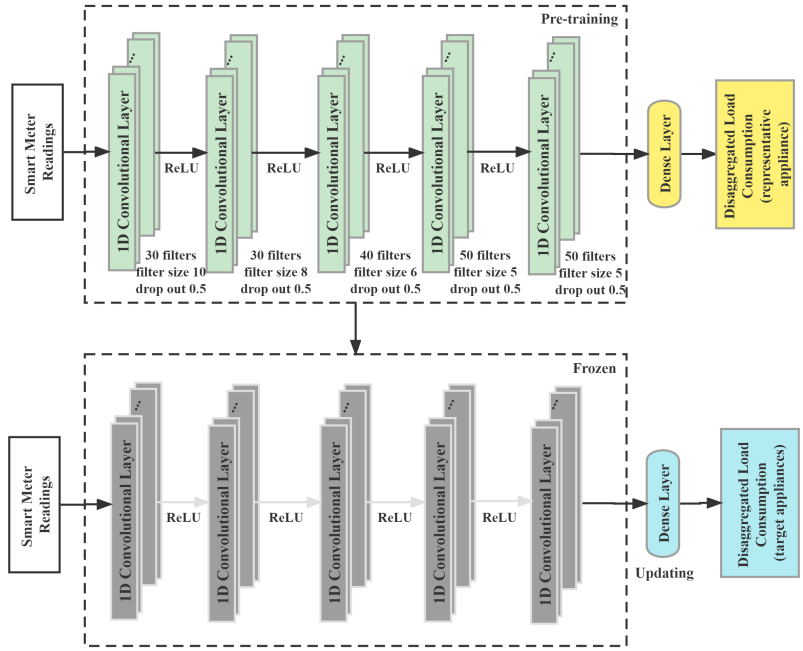


Figure 3.3: Improved NILM model based on sequence-to-point and transfer learning.

3.3.3.2 Principle components analysis

PCA is a mathematical dimensionality reduction method that converts potentially linearly correlated variables into a new set of linearly unrelated variables using orthogonal transformations [172]. PCA is utilized in the model to make the feature more orthogonal and get the new feature.

Specifically, the original disaggregated load consumption matrix of the i-th appliance is

$$(\widehat{l}^{i}_{tp})_{T*P} = \begin{bmatrix} \widehat{l}^{i}_{11} & \cdots & \widehat{l}^{i}_{1P} \\ \vdots & \vdots & \vdots \\ \widehat{l}^{i}_{T1} & \cdots & \widehat{l}^{i}_{TP} \end{bmatrix} \tag{3.5}$$

where T represents the total T time point, and P denotes the features from the disaggregated load consumption of the i-th appliance.

Then, the mean of the disaggregated load consumption of the i-th appliance $\bar{\widehat{l}^i}$ was calculated to eliminate the impact of features with different magnitudes, and the $\widehat{l}^i$ can be written as

$$\widehat{l}^{i*} = \widehat{l}^i - \bar{\widehat{l}^i}. \tag{3.6}$$

Assume Σ is the variance-covariance matrix associated with $\widehat{l}^{i*}$, and Σ is factorable into eigenvalues and eigenvectors $(\lambda_1, e_1), \ldots, (\lambda_P, e_P)$. The principle components can be represented by

$$PC = e_p \widehat{l}^{i*} = e_1 \widehat{l}_1^{i*} + e_2 \widehat{l}_2^{i*} + \cdots + e_P \widehat{l}_P^{i*}. \tag{3.7}$$

Then, based on the obtained components, the goal of PCA is to choose the first n PCs to maximize the variance of these PCs and minimize the variance of the last $(P-n)$ PCs such that

$$\frac{\sum_{i=1}^{n} \lambda_i}{\sum_{i=1}^{P} \lambda_i} \geq \gamma, \tag{3.8}$$

where $\gamma \in [0, 1]$ is a constant to ensure the variance of the unselected PCs sufficiently small, we here chose $\gamma = 0.95$ to get the new features, which result in the final projected two-dimensional features.

3.3.3.3 K-means clustering

K-means is an iterative clustering algorithm that divides the data set into k clusters through similarity index evaluation. K-means initialises k centroids representing k clusters, and each data point is assigned to the nearest cluster. The clustering centroid is then updated as the mean vector of all sample points in the updated cluster, and the optimal clustering result will be found through repeated iteration. Due to that k-means is simple, and the interpretability of its clustering results satisfies the easy-to-distinguish features of the target appliances, after obtaining the new two-dimensional features via PCA, k-means is adopted to cluster the feature set to identify hours in a day when the users are most likely to use each appliance. To use the k-means method, it is necessary to decide on the number of clusters, denoted by k. We followed the elbow method [191] to determine the best value for k, ultimately selecting 2 as the final value.

3.3.4 Evaluation and Discussion

3.3.4.1 Evaluation indicators

For comparison and evaluation purposes, mean absolute error (MAE), normalized signal aggregated error (SAE), and F_1 score are used to evaluate the performance of

the model. Besides, the hit rate (HR) is used to represent the correct classification rate of ADLs pattern recognition.

To test the prediction accuracy at each time point (i.e., the absolute difference between the predicted load $\widehat{l_t^i}$ and the ground truth l_t^i of the i-th appliance), MAE is adopted.

$$MAE = \frac{1}{T}\sum_{t}^{T}|l_t^i - \widehat{l_t^i}| \tag{3.9}$$

In order to evaluate the relative error between the predicted total consumption ($\widehat{l_t^i}$) and the actual total consumption of the i-th appliance (l_t^i), SAE is adopted.

$$SAE = \frac{|l_t^i - \widehat{l_t^i}|}{l_t^i} \tag{3.10}$$

The F_1 score is utilized to verify the accuracy of each appliance's state recognition.

$$Precision = \frac{TP}{TP + FP} \tag{3.11}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.12}$$

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \tag{3.13}$$

where TP (true positive) denotes the sample is positive, and the predicted result is also positive. FP (false positive), predicting the negative sample to be positive. FN (false negative), predicting positive samples to be negative. The precision of a model determines its ability to identify negative samples, while the recall reflects its ability to identify positive samples. The F_1 score is a combination of both metrics, and a higher F_1 score indicates that the model is more reliable. It should be pointed out that the calculation of the F_1 score needs to use a threshold to determine the ON/OFF status of the target appliance. The threshold values of the switching state of the appliances are obtained directly from the description file in the dataset.

In order to verify the accuracy of ADLs pattern recognition, HR is used.

$$HR = \frac{h_c}{24} * 100\% \tag{3.14}$$

where h_c denotes the number of hours of a day that correctly classify the ON/OFF states of an appliance.

3.3.4.2 Data source and preprocessing

We chose UK-DALE [103] as the source data set, and the sampling interval of each sub-meter was 6s. To carry out the comparative experiment, we resampled the data to 8s to align with the data sampling interval in D'Incecco et al. [52]. The UK-DALE dataset consists of five houses in which the kettle, fridge, microwave, washing machine, and dishwasher are commonly used appliances closely related to the ADLs of smart meter users. Motivated by Kelly and Knottenbelt [102], Zhang et al. [234], and Zhong et al. [240], the target devices selected in the model include the kettle, fridge, microwave, washing machine, and dishwasher. Since only houses 1 and 2 in UK-DALE have all these five appliances, we used house 1 as training data to train the improved deep neural network and implement NILM at the appliance level to the unseen house (i.e., house 2 as the test set) via transfer learning.

Abnormal missing data are firstly imputed with mean values. The data are then normalized to facilitate further training of the neural network model.

3.3.4.3 NILM model evaluation

Different experimental environments have different training costs for the same model. In order to verify the training efficiency of the improved neural network, we crop training data into 100000 samples as input to feed the proposed neural network, and the original neural network [52] respectively. Percentages of time and parameters saved by the new model (trained on the washing machine) are shown in Table 3.2.

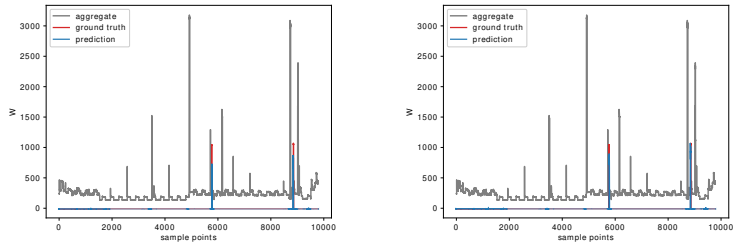
Table 3.2: Number of parameters, training time, and saving percentages of the proposed model compared to the original model

	Original model [52]	Proposed model	Saving percentages
No. of parameters	10,228,249	1,302,139	87.2%
Training time (s)	66.13	27.67	65.7%

It can be seen that the improved model significantly saves the training cost of the original model and improves training efficiency.

The data collected from house 1 in UK-DALE (from November 2012 to November 2013) were used as the training data, and the data from house 2 (from May 20, 2013, to October 10 of the same year) were used as the test data. Fig. 3.4-Fig. 3.8

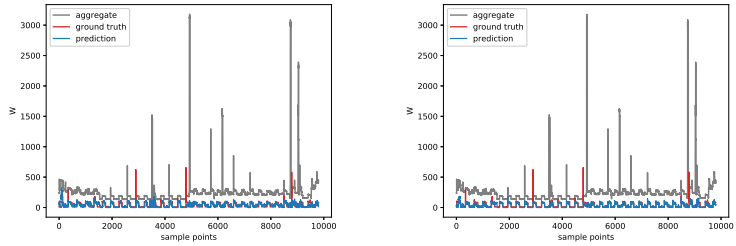
visualize the load disaggregation results of five target devices in the original and the improved models. Table 3.3 shows the training results for the improved and original models.



(a) load disaggregation results of the original model

(b) Load disaggregation results of the proposed model

Figure 3.4: Load disaggregation results for kettle



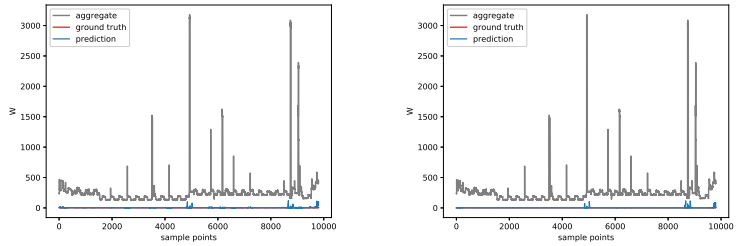
(a) Load disaggregation results of the original model

(b) Load disaggregation results of the proposed model

Figure 3.5: Load disaggregation results for fridge

The above comparative results indicate that the improved model could achieve satisfactory disaggregation with similar accuracy to the original model. In other words, the improved model is competent for load disaggregation on the premise of significantly saving training time.

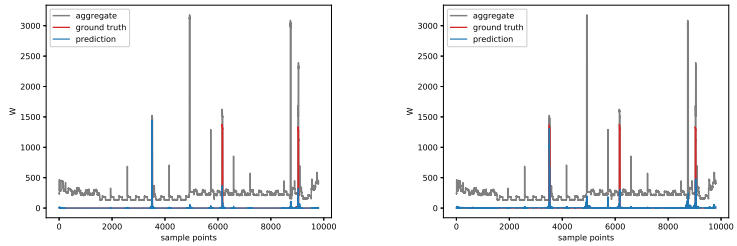
It is worth mentioning that the improved model performs better than the original model in switching devices such as the fridge and kettle. MAE of the fridge by the proposed model decreased from 25.10 to 20.36 of the original model. The visualization results also show that the fitting shape of the proposed model for the fridge and kettle is better than that of the original model. On the other hand, more



(a) Load disaggregation results of the original model

(b) Load disaggregation results of the proposed model

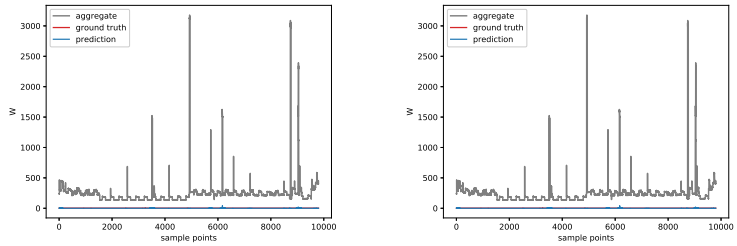
Figure 3.6: Load disaggregation results for dishwasher



(a) Load disaggregation results of the original model

(b) Load disaggregation results of the proposed model

Figure 3.7: Load disaggregation results for microwave



(a) Load disaggregation results of the original model

(b) Load disaggregation results of the proposed model

Figure 3.8: Load disaggregation results for washing machine

Table 3.3: Test results for the proposed model and original model; Best results are marked in bold

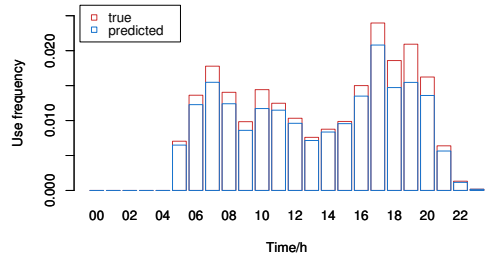
	Proposed model			Original model		
	MAE	SAE	F_1	MAE	SAE	F_1
Kettle	**9.90**	0.05	**0.91**	10.21	**0.03**	0.86
Fridge	**20.36**	0.26	**0.81**	25.10	**0.21**	0.71
Dishwasher	23.45	0.46	**0.69**	**12.87**	**0.03**	0.53
Microwave	14.48	0.98	**0.53**	**7.14**	**0.35**	0.52
Washing machine	7.17	**0.35**	0.73	**7.12**	0.45	**0.80**

accurate disaggregation results were obtained from the original model for finite-state devices with more complex operation statuses, such as dishwashers, washing machines, and microwaves. For the state recognition of each appliance, the improved model is superior to the original model on most of the target appliances (see F_1 metrics).

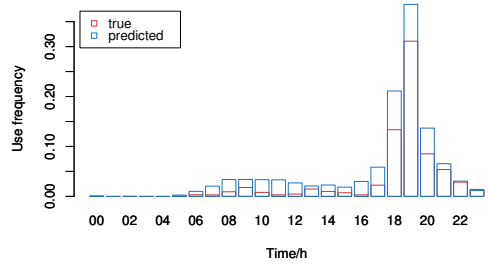
3.3.4.4 ADLs pattern recognition evaluation

The dataset we used allows us to calculate the power consumption of the target appliance in hours. To detect ADLs of the smart meter users, we calculate the mean power consumption and the frequency of power consumption in each hour for each appliance based on the predicted load disaggregation results. Fig. 3.9 shows the predicted and true use frequency of the dishwasher and microwave. The frequency of use is calculated by the ratio of the number of days that the appliance has been used in that hour to the total number of days considered. The mean power and usage frequency are taken as two features to conduct PCA, and then k-means is applied to cluster the new feature set, thus inferring the time of a day when the occupant is most likely to use the target appliance. Table 3.4 lists the HR of five target appliances for house 2. To better demonstrate the ADLs pattern recognition performance, Fig. 3.10 compares the inferred cluster results of dishwasher and

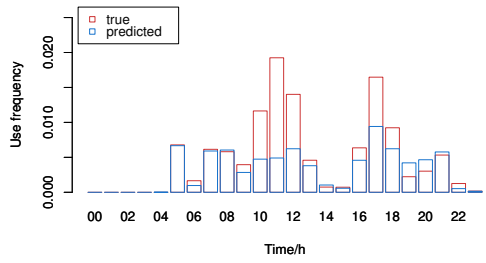
microwave with the clustering results based on ground truth data. It can be easily found that the dishwasher (with an HR of 100%) is most likely to be used around dinner time.



(a) Kettle

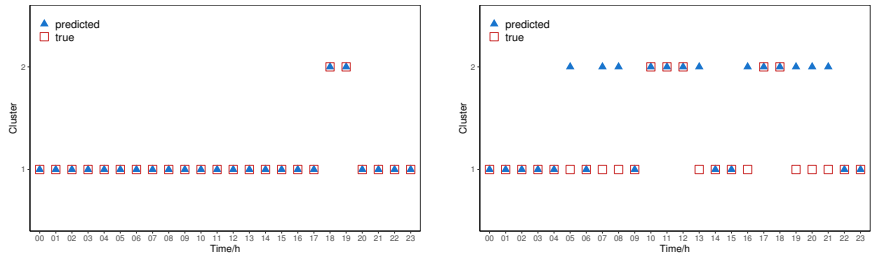


(b) Dishwasher



(c) Microwave

Figure 3.9: Usage frequency comparison of kettle, dishwasher and microwave



(a) Dishwasher (b) Microwave

Figure 3.10: ADLs pattern recognition for dishwasher and microwave: cluster 1 represents the OFF status and cluster 2 represents the ON status

Table 3.4: Hit rate for the target appliances

	HR
Kettle	100.00%
Fridge	75.00%
Dishwasher	100.00%
Microwave	66.67%
Washing machine	95.83%

It can be seen from Table 3.4 that the improved network model can accurately infer the ADLs pattern of participants using the kettle, dishwasher, and washing machine. However, for the fridge and microwave, the inference results still have room to be improved.

As aforementioned, the proposed model produced less accurate disaggregation results for dishwashers and microwaves. As shown in Fig. 3.9, compared with the kettle, the predicted use frequency results of both the dishwasher and microwave deviated significantly from the ground truth. However, it is interesting to note that the HR for the dishwasher reached 100%, whereas the microwave only achieved an HR of 66.67%. One possible reason for this could be that although the predicted use frequency for both the dishwasher and microwave was mediocre compared with that of the kettle, the overall trend of the dishwasher was similar to the ground truth. Instead, the predicted use frequency for microwaves differed much from the ground truth. Therefore, the ADLs pattern recognition model may produce less accurate cluster results for microwave usage than the dishwasher, as seen from Fig. 3.10.

Overall, the proposed model is useful for performing ADLs pattern recognition based on NILM. Moreover, the obtained results based on ADLs pattern recognition can be used as inputs and prior knowledge to abnormal detection algorithms. In this way, abnormal behaviour of the appliance can be monitored, and potential risks can be detected in time, which will be further investigated in our future work.

3.4 Individual Level and Higher Aggregation Level load Forecasting: A Consensus-based load Profiling and Forecasting System

3.4.1 Framework Design Overview

In the proposed system, we consider real-world smart meter data collected from buildings in Cardiff, UK, where the load for different buildings in different seasons vary. The data used in this model are collected from Cardiff Council [1], which includes different types of buildings such as office buildings, community facilities, schools, and cultural buildings. Data on historical load were collected every half hour, and missing values below 20% were replaced by column means. We removed other missing data since they may adversely affect forecasting accuracy during the model training phase. The pre-processed data of each building includes half-hourly load records in 2015 and physical information such as building type, number of floors, energy rating certificate, and heating types. The detailed physical informa-

tion of the considered buildings is shown in Table 3.5.

Table 3.5: Physical information of the considered 169 buildings.

Building Type	**Count**	**Floor Number**	**Count**	**Heating Type**	**Count**	**Energy Rating Certificate**	**Count**
Primary Schools	63	0	6	Gas	158	A	2
Community Facilities	29	1	51	Biomass	1	B	5
High Schools	21	2	75	Electricity	1	C	42
Leisure and Sports	11	3	21	Oil	1	D	46
Care Services Buildings	10	4	9	NA	8	E	22
Workshops and Depots	10	5	3			F	5
City Services	7	6	1			G	6
Key and Cultural	6	7	1			N	37
Parks Buildings	6	NA	2			NA	4
Core Offices	6						

Most buildings are educational, with at most three floors and predominantly gas heating. Most of the buildings are on the mid-level of energy rating (C to E). Buildings with different physical characteristics may display different electricity consumption patterns. Analysing the relationship between physical factors and load usage patterns can assist grid operators in optimizing building energy management strategies. The proposed system aims to understand the relationship between physical characteristics and load usage behaviour of buildings, as well as to develop customized load forecasting models for building groups that display different load usage patterns. The system also aims to investigate the relationship between different forecasting algorithms and load consumption patterns at different data aggregation levels.

A graphical overview of the system framework is shown in Figure. 3.11 and explained step-by-step in subsections 3.4.2. Specifically, the proposed system framework contains four layers. Layer 1 segments the historical building smart meter data into seasonal time series and normalizes the data to prepare it for unsupervised modelling. After this, layer 2 conducts the consensus-based robust clustering to cluster the buildings with similar load usage patterns in different periods. Then, layer 3 analyzes the clustering profiles, including their relationship patterns with

the building physical characteristics and the dynamic load usage profiles trajectory tracking route. By analysing cluster results, we can identify key drivers that affect building load consumption in different clusters, which can help to develop customized load forecasting models for each building cluster at layer 4. Specifically, to assist the individual level forecasting, feature selection was adopted to enhance the model performance further. Then, layer 4 explores the relationship between the performance of different forecast algorithms and consumption patterns at both the clustered building and aggregation levels. In the final stage, consensus-based model training will be conducted to select the most appropriate forecasting models for smart meter data on both levels.

3.4.2 Implementation

3.4.2.1 Building smart meter data management

To understand the typical characteristics of the building smart meter data, we first briefly analyze the load usage patterns associated with different physical building factors in different seasons. This study uses load consumption data from 169 Cardiff buildings in 2015, and the average daily load consumption regarding physical information of the considered buildings is summarized in Figure 3.12-3.15.

As observed, core offices, high schools, key and cultural buildings, and leisure and sports buildings consume the most significant amount of electricity throughout the year. The peak load time for most buildings is between 10:00 and 14:00, the busiest working hours. In contrast, buildings that provide care services, community facilities, parks, primary schools, workshops, and depots use less electricity throughout the year. During spring and summer, the city services buildings consume more electricity in the mornings and at night, while they consume very little electricity in the autumn and winter.

In addition, buildings that have fewer floors consume less electricity, whereas buildings with more floors consume more. In addition, buildings heated by biomass consume the most considerable amount of electricity throughout the year. Other heating methods, including electricity, gas, and oil, consume much less load when compared to buildings heated by biomass. It is noteworthy that, unlike other heating types, which consume more electricity during peak hours in the cold season, buildings heated by electricity have lower load consumption during winter peak hours. This could be explained by the fact that, for buildings with electricity as the primary heating source, there are usually storage heaters/hot water tanks alongside to fully take advantage of time-of-use electricity prices (e.g., Economy 7 in the UK) where radiators/water tanks are heated and stored during the off-peak periods

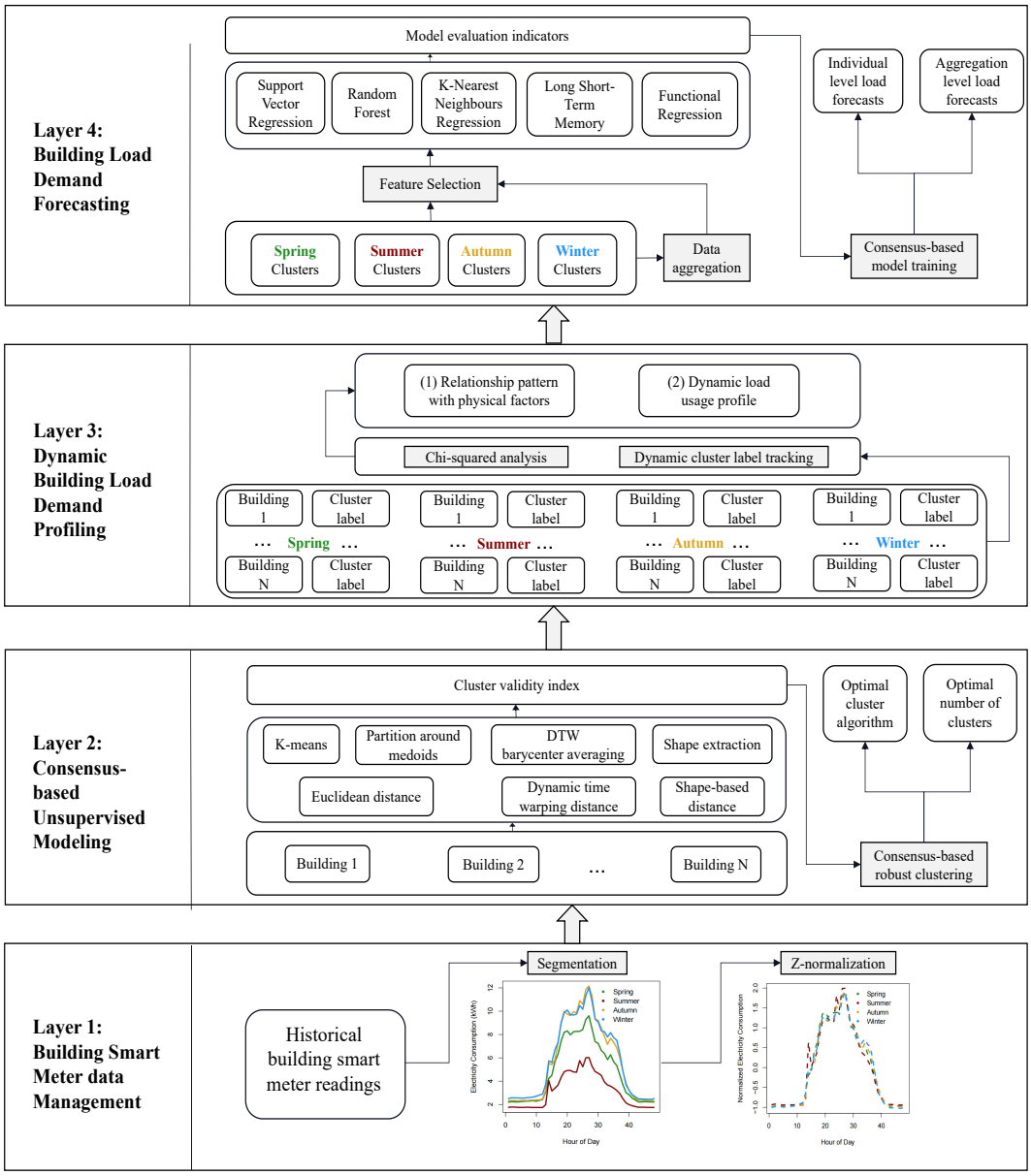


Figure 3.11: The framework of the consensus-based load profiling and forecasting system.

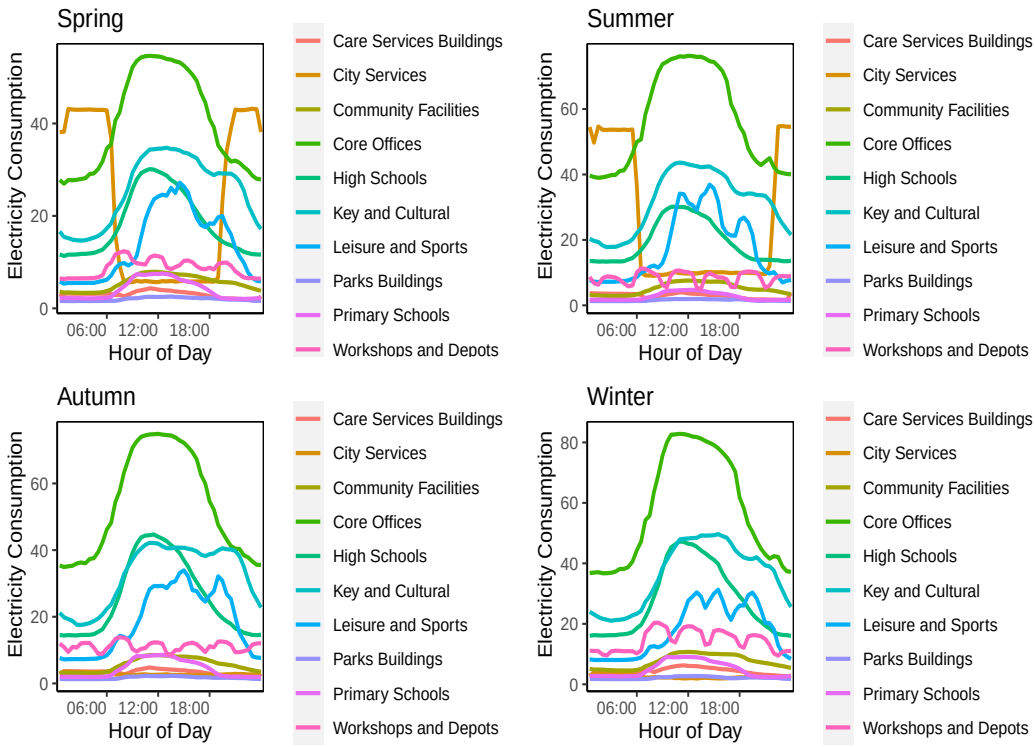


Figure 3.12: Load consumption of different types of buildings in four seasons.

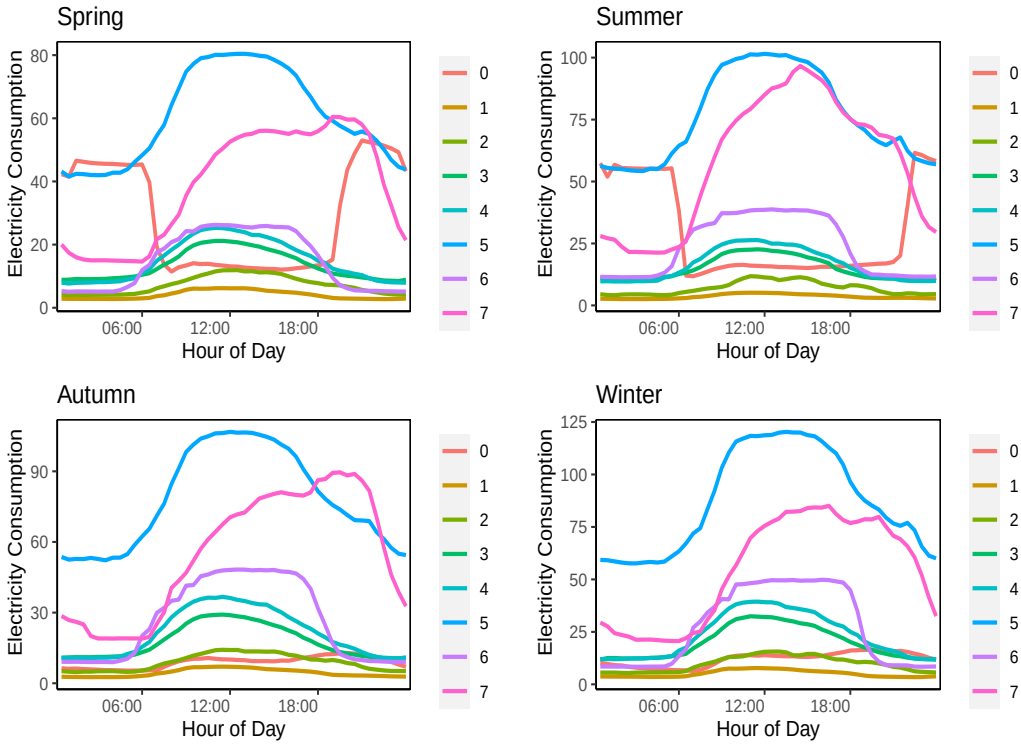


Figure 3.13: Load consumption of different number of floors in four seasons.

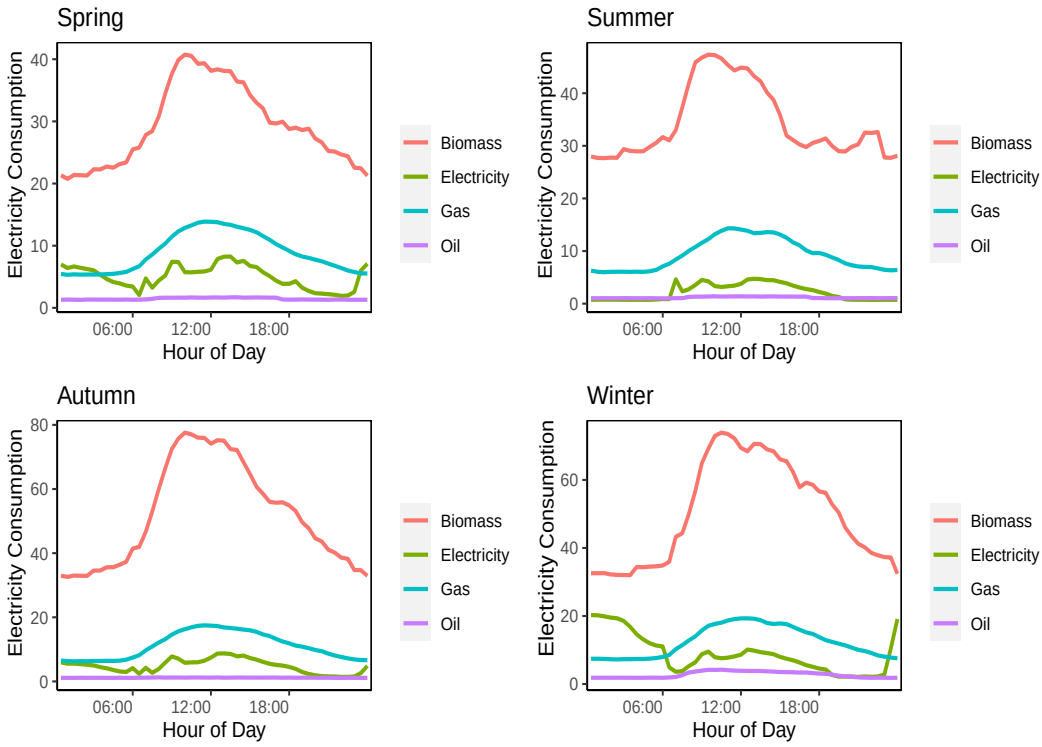


Figure 3.14: Load consumption of different heating types of buildings in four seasons.

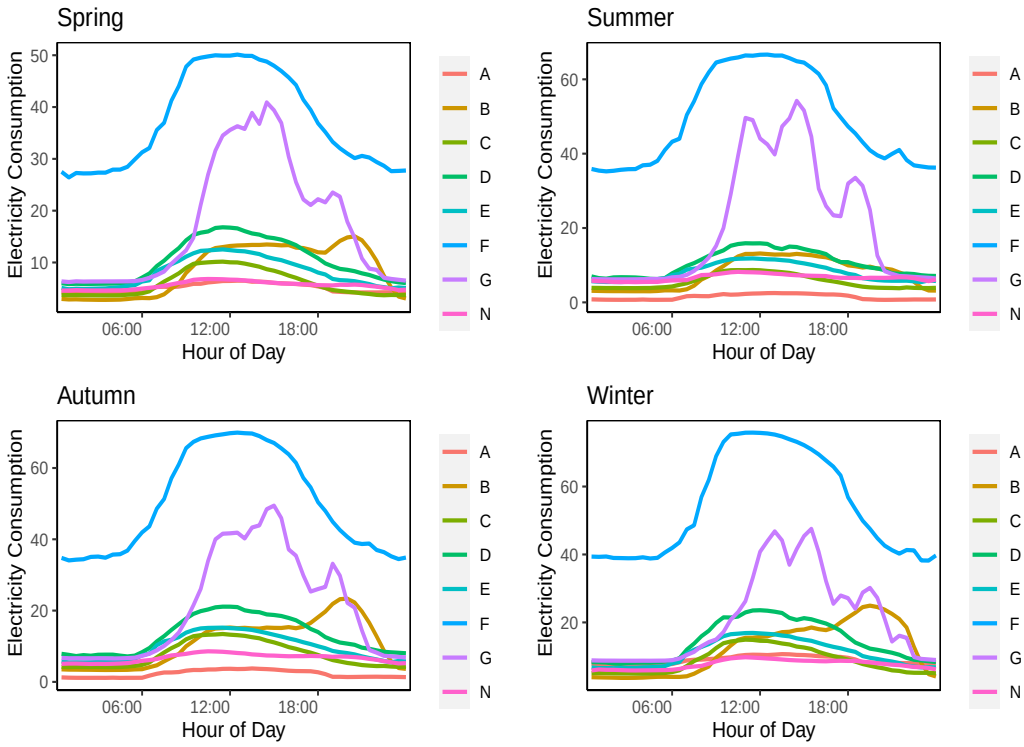


Figure 3.15: Load consumption of different building energy rating certificate in four seasons.

(usually from around midnight to early morning) and are used later (e.g., during daytime), thus resulting in different peak load periods between buildings heated by electricity and those heated by other energy sources during winter.

Based on the seasonal segmentation, we aim to model the load profiles of the buildings in layer 2 using unsupervised learning techniques. However, these algorithms must deal with scale and translation invariance to prioritize the shape features of the load patterns over amplitude ones, especially for time series clustering [98, 50, 211]. Therefore, z-normalization was used to normalize the load profiles:

$$Z_j = \frac{x_j - \mu_j}{s_j} \tag{3.15}$$

where Z_j are the normalized values calculated for all load profiles x_j for season j. μ_j and s_j represent the mean feature value and the standard deviation, respectively. We use the mean feature value and the standard deviation over the season months rather than the whole year for the seasonal load profiles. It is important to use this segmented normalization because we are focusing on dividing offices by seasons. This will enable us to determine the dynamic trajectory of the load usage behaviour in different seasons.

To assess the suitability for clustering before and after the z-normalization, we then used a function `get_clust_tendency` from the `factoextra` library in `R`. Table 3.6 presents Hopkins statistical scores for the raw and z-normalized data in the four seasons. The z-normalized data generates the highest scores for all four seasons, indicating that more meaningful clusters are present in the data set after z-normalizing.

Table 3.6: Frequency analysis of the physical information for the simulated offices.

	Cluster tendency (Hopkins statistic)			
	Spring	Summer	Autumn	Winter
Raw data	0.95	0.96	0.96	0.98
Z-normalized data	**0.98**	**0.98**	**0.99**	**0.99**

3.4.2.2 Consensus-based unsupervised modelling

In layer 2, we perform clustering on the normalized seasonal load records segments. Clustering algorithms must take into account two key components: distance mea-

surements and prototyping functions [148]. It has been demonstrated that clustering results can be unstable when only one clustering algorithm is used [92, 184]. For this reason, we employ consensus-based robust clustering across multiple distance measurements and prototype combinations , which uses three cluster validity indices to select the optimal cluster algorithm by majority voting. Table 3.7 lists the clustering algorithm combinations used in the proposed system.

The Euclidean distance [173], which approximates a one-to-one correspondence between each pair of sequences, is the most commonly used distance measurement. On the other hand, time series distance measurements must be invariant to specific distortions of the data to yield accurate results, which has been regarded as a limitation of the Euclidean distance [173]. Therefore, except for using the Euclidean distance, two state-of-the-art distance measurements, namely the dynamic time warping distance (DTW) [19] and the shape-based distance (SBD) [160] also adopted in the model to produce more accurate similarity measurements for the load profiles.

To quantitatively choose the optimal number of clusters and also evaluate the clustering results, three cluster validity indices (CVIs) are used in the validation process to produce robust validation results: The silhouette index, Davies-Bouldin (DB) index, and Calinski-Harabasz (CH) index. These metrics simultaneously measure the cohesion of the objects within clusters and the separation between clusters. For the Silhouette index, it is obtained by contrasting the average distance of objects within the same cluster with the average distance to objects in other clusters where it takes values between -1 and 1. Cluster configurations with a higher Silhouette index value are considered to be more optimal. Similarly, a higher CH index value signifies a better clustering result. In contrast, for the DB index, a lower value indicates a better clustering configuration. A more detailed description of the evaluation metrics can be found in [46] and [183].

3.4.2.3 Dynamic building load profiling

After the consensus-based unsupervised modelling, a dynamic clustering structure is adopted in layer 3 to capture seasonal/dynamic changes of load usage behaviours in different buildings, which is inspired by de Zepeda et al. [46]. The proposed dynamic building load profiling structure is illustrated in Figure 3.16.

Under the dynamic profiling structure, individual consensus-based clustering analysis is implemented for each season. The ultimate clustering results through the dynamic clustering structure consists of two parts: (1) relationship pattern between the clusters and the physical factors of the buildings. (2) dynamic load usage behaviour trajectories of each building.

Table 3.7: List of the used combination of the consensus-based clustering algorithm. DTW: dynamic time warping, SBD: shape-based distance; PAM: partition around medoids; DBA: DTW barycenter averaging.

Distance measurements & Equations	**Cluster prototypes**	**Description**
Euclidean Distance: $d_{ecul}(y_i^a, c_k) = \sqrt{\sum_{i=1}^{24}(y_i^a - c_k)^2}$	K-means	Random initialise K cluster centers, then the Euclidean distance between the load vector y_i^a and the cluster center c_k is minimized. For each cluster k, update the cluster center c_k to be the mean attribute vector of all observations in cluster c_k.
	PAM	Random initialise K cluster centers, then the Euclidean distance between the load vector y_i^a and the cluster center c_k is minimized. For each cluster k, update the cluster center c_k by finding the representative load vector y_i^a that minimizes the Euclidean distance to other vectors in this cluster.
DTW: $d_{dtw}(y^a, y^b) = \frac{\min\sqrt{\sum_{p=1}^{P}\omega_p}}{P}$, $\omega_p = (y_i^a - y_l^b)^2 \in [1:24] \times [1:24]$	DBA	Random initialise K cluster centers, then the DTW distance between the load vector y_i^a and the cluster center c_k is minimized. For each cluster k, update the cluster center c_k by finding the representative load vector y_i^a that minimizes the DTW distance to other vectors in this cluster.
SBD: $d_{sbd}(y^a, c_k) = 1 - \frac{\max(NCCc(y^a, c_k))}{\|y^a\|_2 \|c_k\|_2}$	Shape extraction	Random initialise K cluster centers, and then update the cluster center c_k by finding the maximized normalized cross-correlation based on the SBD distance.

Where:
$y^a = (y_1^a, y_2^a, \ldots, y_i^a, \ldots, y_{24}^a)$ and $y^b = (y_1^b, y_2^b, \ldots, y_l^b, \ldots, y_{24}^b)$ are two daily load curves;
for d_{dtw}, $\omega_p \in \{\omega_1, \ldots, \omega_p, \ldots, \omega_P\}$ is defined as the warping path and satisfying the following three conditions:

- *Boundary condition*: $\omega_1 = (1, 1)$ and $\omega_P = (24, 24)$.
- *Monotonicity condition*: $i_1 \leq i_2 \leq \ldots \leq i_P$ and $l_1 \leq l_2 \leq \ldots \leq l_P$.
- *Step size condition*: $\omega_{p+1} - \omega_p \in \{(1, 0), (0, 1), (1, 1)\}$ for $p \in [1 : P - 1]$.

for d_{sbd}, $NCCc$ is the cross-correlation with coefficient normalization sequence between the load curve y^a and the cluster centroid c_k, and $\|\cdot\|_2$ is the l_2 norm.

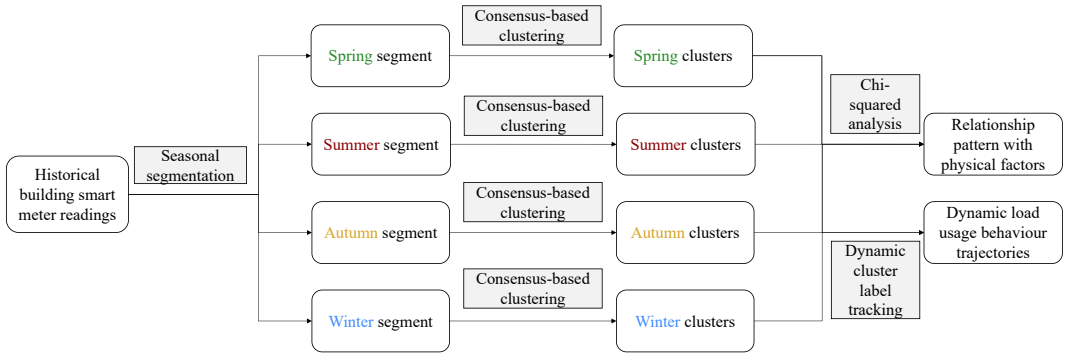


Figure 3.16: Dynamic analysis to capture seasonal change.

Specifically, the clusters for different seasons are described by testing the difference in the building physical factors. Since the physical factors (detailed in the former Table 3.5) can all be seen as the categorical variables, the chi-square test [161] is adopted to analyze the association between the clusters and the physical factors. Then, based on the cluster results for each season, the dynamic load usage behaviour patterns for each building can be tracked, which is a cluster trajectory for each building (e.g., a cluster assignment for each building under each season). More importantly, by analysing the dynamic cluster trajectories, typical dynamic clustering trajectories over the year corresponding to the physical factors possessed by the building can be sketched by category summary. We highlight that the proposed dynamic building load profiling layer not only allows the investigation of different load usage behaviours but also facilitates the understanding of seasonal behavioural changes of the smart meter users.

3.4.2.4 Building load forecasting

Each building has a unique electricity usage pattern due to distinct physical information and residential behaviours. Therefore, it is desirable to choose appropriate forecast algorithms for different consumption patterns. Based on the clustering results in layer 3, layer 4 aims to build personalized forecasting models for different cluster groups by consensus-based model training strategy and explore the relationship between the performance of forecast algorithms and different consumption patterns at both the building level and the aggregation level. Specifically, different algorithms are used to build forecasting models, and the consensus-based model training strategy is to choose the optimal forecasting model with the best perfor-

mance. Figure 3.17 shows the overall consensus-based model training flow.

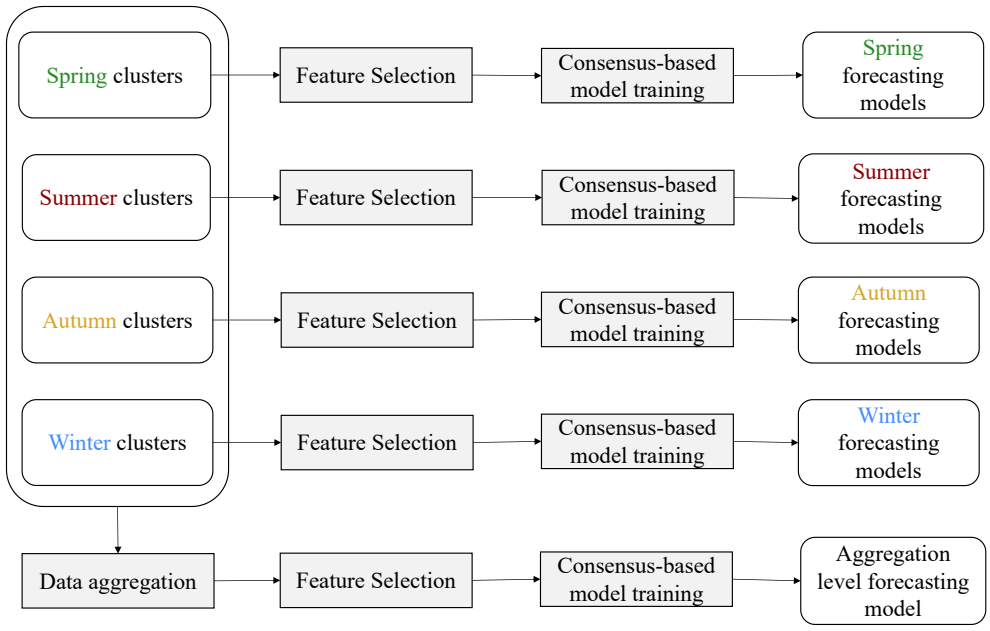


Figure 3.17: Consensus-based model training process at the building level and the aggregation level.

In this layer, the relationship between the performance of forecast algorithms and different consumption patterns at both building and aggregation levels are explored. Specifically, the influential load consumption-related features are determined and extracted from the clustered data; then RF is adopted to produce the feature importance distribution for both the building level clusters and aggregation level load data, and the five most significant features are then selected to train the forecasting models at both levels. After the feature selection, five algorithms (SVR, RF, KNR, LSTM, functional regression) are applied to build forecast models, with grid search being used to find the optimal combination of parameters for each model:

- **Support Vector Regression**

  SVR has been widely applied in energy forecasting applications for its high effectiveness in solving non-linear problems [8]. SVR adopts the structural risk minimization principle, which minimizes the training error and the upper bound of the generalisation error [53]. Given a training dataset $T = (x_1, y_1), ..., (x_i, y_i), ..., (x_N, y_N)$, where $x_i \in R^m$ denotes the i-th observation

Algorithm 1: Random forest for regression

for *each* $b \in [1, B]$ **do**
  Select a bootstrap sample $\mathbf{Z}^*$ of size N from the training dataset;
  Set a minimum node number $node_{min}$, and grow a random forest tree T_b to the bootstrapped data;
  repeat
    Choose r variables randomly from the input variables select the best split point from r;
    Split the node into two sub-nodes;
  until reach $node_{min}$
end
Output: fully grown forest $\{T\}_1^B$
To predict at a new point x:

$$\hat{y} = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \tag{3.17}$$

which is a m-dimensional input vector, $y_i \in R$ is the output corresponding to x_i, and N denotes the size of training set. For non-linear SVR, the basic idea is to introduce a kernel function to efficiently map the input space into a higher dimensional feature space, in which the problem becomes linearly separable [117]. The decision function of SVR can be represented in Eq. (3.16)

$$\hat{y} = \langle w, \phi(x) \rangle + b \tag{3.16}$$

where $\phi(x)$ is the hypothetical higher dimensional feature space. Coefficients w and b need to be estimated based on the structural risk minimization principle.

A main advantage of SVR is that the loss function penalizes deviations greater than a threshold, which often leads to the sparse representation of the decision rule, thus bringing major algorithmic and representational strength [175]. Hence in our paper, we selected SVR as one of the regression algorithms to perform energy forecasting.

- **Random Forest**

  RF [23] adopts the random method to establish a forest, which is an ensemble method. RF is often used for classification and regression problems. The RF for regression is detailed in Algorithm 1 [179].

  In Algorithm 1, recall the training dataset of size N, we build B regression trees on bootstrapped training samples of size N. When a split in a tree is considered, a random sample of r variables is selected as split candidates from the full set of m variables. To predict a new point, it uses the average output of B regression trees; see Eq. (3.17).

RF achieves high prediction accuracy without increasing computation time. Moreover, RF is a robust algorithm when applied to data with missing values or imbalanced data [23]. Therefore, we consider RF one of the comparison algorithms in layer 4.

- **K-Neighbours Regression**

  The KNR algorithm selects the nearest k load samples $y_1, ..., y_i, ..., y_k$ by computing the distance between the observed load samples y and other load samples in the feature space. y_i represents the i-th closest load sample to the load samples y. Then, the forecasted load $\hat{y}$ in the KNR can be calculated in Eq. (3.18)

  $$\hat{y} = \frac{1}{k} \times \sum_{1}^{k} \hat{y}_{y_i} \tag{3.18}$$

  where $\hat{y}_{y_i}$ denotes the i-th closest load sample to $\hat{y}$. The forecasted load $\hat{y}$ is calculated by averaging the forecasted values of its k nearest neighbours on the assumption that each neighbour contributes uniformly to the forecast.

  As an easy-to-implement theoretical tool, KNR is frequently employed to solve nonlinear problems. In particular, the algorithm is not sensitive to outliers and has higher forecasting accuracy without requiring assumptions about the collected data [20]. Therefore, KNR is selected as one of the comparison algorithms in layer 4.

- **Long Short-Term Memory** LSTMs [85] are deep recurrent neural networks widely employed to model time series data, which perform well when the time series data have implicit temporal dependencies. Unlike classic ANNs, LSTMs use activations from previous time steps as inputs for the current prediction. In this way, the recurrent connection enables the model to develop a memory for past events embedded in its hidden state variables. Assume that (x_t, y_t) is input at time t, h_{t-1} is the output in the previous time step, and C_{t-1} is the memory cell in the previous time step. The output of the current network can be calculated in Eq. (3.19)

  $$h_t = f(h_{t-1}, x_t) \tag{3.19}$$

  where $f(\cdot)$ is the recurrence function, in which the input gate, output gate, and forget gate (for gradient vanishing and explosion problems) are introduced.

  New information from the input is updated to the cell state by the input gate, irrelevant information is removed from the h_{t-1} state by the forget gate, and

the output gate will decide the output h_t. Long-term and short-term sequence-related information is learned and memorized by these gates simultaneously, while memory cell C stores the state information.

LSTMs can also be configured with multi-sequence inputs, allowing them to capture information across various timescales [123]. Thus, LSTM is included in layer 4 as a training model.

- **Functional Regression** FR models transform infinite-dimensional load data into finite-dimensional functions while maintaining the inherent continuity of the original data. We consider the FR as one of the forecasting algorithms in layer 4, which considers the load records of the last week and last day as functional predictors to predict the functional load curve of the current day:

$$\hat{F} = \mathcal{B}_0 + \sum_{i=1}^{N} (\mathcal{B}_i \times F_i) + \epsilon_i \tag{3.20}$$

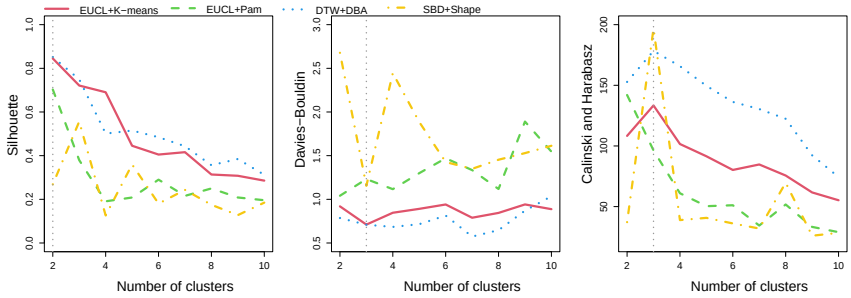
where F denotes the functional representation of the original load training dataset T, which can be derived using basis functions such as the splines, Fourier series, and principle components [170]. Our study uses the splines to represent load data as functional curves since they effectively model smooth functions [105]. $\mathcal{B}$ represents the coefficient function, and the ϵ denotes the residual error.

Each model aims to perform day ahead single-step prediction load demand forecasting. The forecasting model with the best performance for each cluster and the aggregated load data will be selected as the final personalized forecasting model. This allows us to further compare the performance of aggregation level forecasts with building level forecasts to explore the effect of the data aggregation process on forecasting performance.

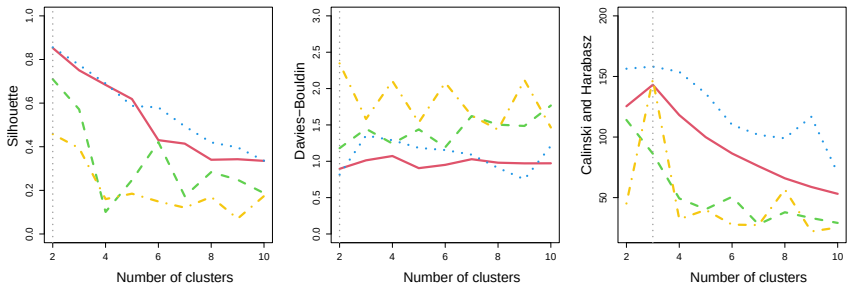
3.4.3 Evaluation and Discussion

3.4.3.1 Cluster validity results

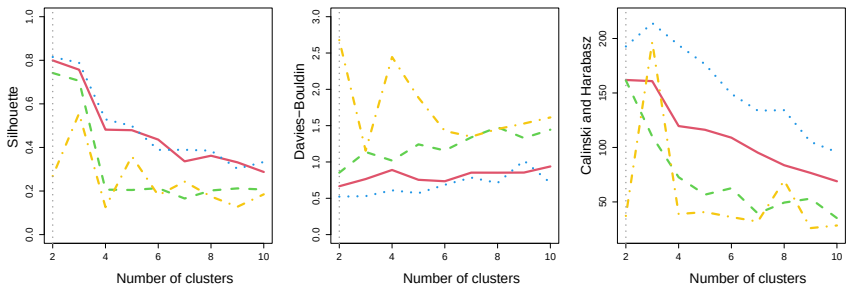
We adopt consensus-based clustering to cluster the 169 Cardiff buildings based on their daily load records in different seasons. The optimal cluster number and algorithm for each season are chosen according to the majority vote from the CVIs. We consider cluster numbers ranging from 2 to 10, i.e., $K = 2, \ldots, 10$. Fig. 3.18 plots the visualizations of the CVIs for the cluster results in each season.



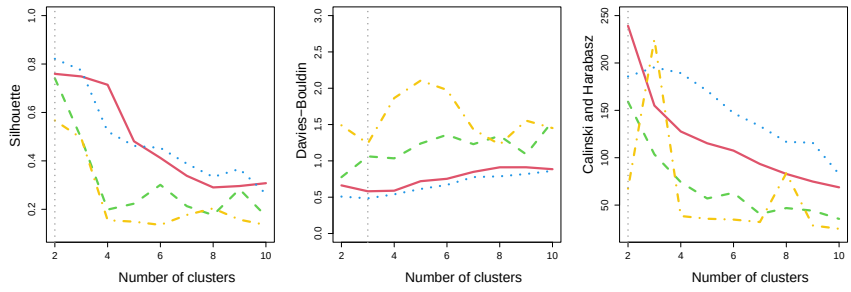
(a) CVIs for spring season.



(b) CVIs for summer season.



(c) CVIs for autumn season.



(d) CVIs for winter season.

Figure 3.18: A comparison of the CVIs of consensus-based clustering in different seasons. The grey vertical dashes line indicates the optimal cluster number for each CVI.

Based on the majority vote, the CVIs vote $K = 3$ as the best clustering number for the spring season and $k = 2$ for the other three seasons. The final chosen clustering algorithms and their corresponding CVIs for the four seasons are listed in Table 3.8.

Table 3.8: Lists of the optimal clustering algorithms and their corresponding CVIs for the four seasons.

Season	**Optimal clustering algorithm**	**Optimal cluster number**	**CVIs**		
			Silhouette	**DB**	**CH**
Spring	DTW+DBA	3	0.75	0.71	178.38
Summer	DTW+DBA	2	0.86	0.82	156.46
Autumn	DTW+DBA	2	0.82	0.52	192.69
Winter	DTW+DBA	2	0.82	0.51	185.74

Dynamic time warping clustering is selected as the optimal clustering method for all four seasons. Fig. 3.19 plots the cluster centroids for the clusters in the four seasons.

3.4.3.2 Relationship pattern between seasonal load clusters with building physical factors

Based on the consensus results, the chi-square test is then performed to describe the difference between clusters for the categorical physical factors of the building in different seasons. Table 3.9-3.12 gives the test results for the four building physical variables.

Building type and the number of floors in all four seasons were statistically significantly associated with clusters based on the chi-square test. In the cold seasons, i.e., the autumn and the winter, the heating type was found to be statistically significantly associated with the clusters, which indicates that the increased use of heating appliances in the cold seasons is the predominant factor that impacts the building load consumption. Meanwhile, the energy rating certificate was found to be statistically significantly associated with clusters in the spring and summer, whereas the heating type did not correlate statistically significantly with clusters. Based on this, it was demonstrated that building load is primarily determined by

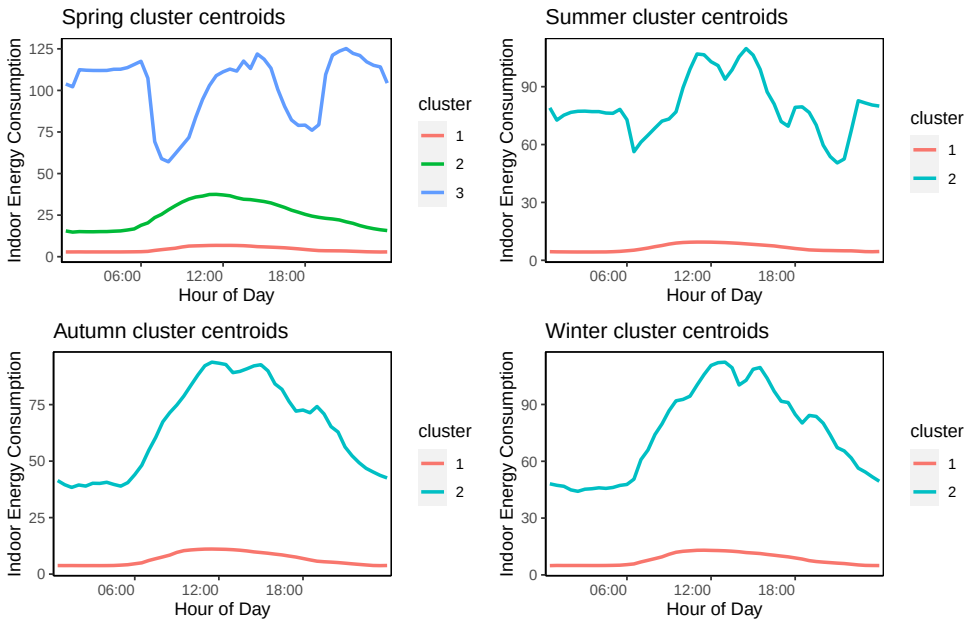


Figure 3.19: The cluster centroids of clusters in different seasons.

the type of building services and the number of floors.

Furthermore, the highest percentage and number of associated categories are selected together with the mean, min, and max load usage to form the cluster label for each cluster, which are summarized in Tables 3.13 for each of the four seasons to facilitate tracking of the dynamic cluster trajectory for each building. By combing both the clustering results and the corresponding building physical information, we can conclude that each cluster has distinct characteristics as described in the cluster label.

Clustering is associated with building types, floor numbers, and energy rating certificates in spring and summer. During the spring season, primary schools are the main building types in cluster 1 for all four seasons. Primary schools are the most common building types in cluster 1, which has buildings with fewer floors and high energy ratings during the spring. Buildings in cluster 3, on the other hand, have more floors and consume more electricity during the spring. Cluster 2 can be seen as a transition cluster between cluster 1 and cluster 3, with medium load consumption and medium energy rating certificates.

In addition, clusters in the summer, autumn, and winter show more distinct characteristics than those in the spring. Fewer floor numbers and a lower load con-

Table 3.9: Chi-square test results for spring clusters.

Variable	**Category**	**Clusters**			**Test results**	
		Cluster 1	Cluster 2	Cluster 3	x^2	p
Building Type	Care Services Buildings	10	0	0	115.195	0.000***
	Community Facilities	27	2	0		
	Core Offices	2	2	1		
	High Schools	5	16	0		
	Key and Cultural	1	5	0		
	Leisure and Sports	9	1	1		
	Parks Buildings	6	0	0		
	Primary Schools	63	0	0		
	Workshops and Depots	9	1	0		
Number of Floors	0	1	1	0	67.285	0.000***
	1	47	2	0		
	2	67	7	1		
	3	12	9	0		
	4	4	5	0		
	5	1	1	1		
	6	0	1	0		
	7	0	1	0		
Heating Type	Biomass	1	0	0	5.414	0.492
	Electricity	0	1	0		
	Gas	26	130	2		
	Oil	0	1	0		
Energy Rating Certificate	A	2	0	0	35.967	0.001***
	B	5	0	0		
	C	37	5	0		
	D	33	13	0		
	E	18	4	0		
	F	3	1	1		
	G	4	1	1		
	N	30	3	0		

$***p$ <0.01, $**p$ <0.05.

Table 3.10: Chi-square test results for summer clusters.

Variable	**Category**	**Clusters**		**Test results**	
		Cluster 1	Cluster 2	x^2	p
Building Type	Care Services Buildings	10	0	33.684	0.000***
	Community Facilities	29	0		
	Core Offices	3	2		
	High Schools	20	1		
	Key and Cultural	4	2		
	Leisure and Sports	10	1		
	Parks Buildings	6	0		
	Primary Schools	63	0		
	Workshops and Depots	9	1		
Number of Floors	0	2	0	33.277	0.000***
	1	49	0		
	2	73	2		
	3	19	2		
	4	8	1		
	5	2	1		
	6	1	0		
	7	0	1		
Heating Type	Biomass	1	0	0.139	0.987
	Electricity	1	0		
	Gas	151	7		
	Oil	1	0		
Energy Rating Certificate	A	2	0	21.723	0.003***
	B	5	0		
	C	42	0		
	D	43	3		
	E	21	1		
	F	3	2		
	G	5	1		
	N	33	0		

$***p$ <0.01, $**p$ <0.05.

Table 3.11: Chi-square test results for autumn clusters.

Variable	**Category**	**Clusters**		**Test results**	
		Cluster 1	Cluster 2	x^2	p
Building Type	Care Services Buildings	10	0	26.563	0.001***
	Community Facilities	28	1		
	Core Offices	3	2		
	High Schools	17	4		
	Key and Cultural	4	2		
	Leisure and Sports	9	2		
	Parks Buildings	6	0		
	Primary Schools	63	0		
	Workshops and Depots	9	1		
Number of Floors	0	1	1	29.361	0.000***
	1	49	0		
	2	71	4		
	3	17	4		
	4	8	1		
	5	2	1		
	6	1	0		
	7	0	1		
Heating Type	Biomass	1	0	12.633	0.006***
	Electricity	0	1		
	Gas	11	147		
	Oil	0	1		
Energy Rating Certificate	A	2	0	9.887	0.195
	B	5	0		
	C	40	2		
	D	42	4		
	E	21	1		
	F	3	2		
	G	5	1		
	N	31	2		

$***p$ <0.01, $**p$ <0.05.

Table 3.12: Chi-square test results for winter clusters.

Variable	**Category**	**Clusters**		**Test results**	
		Cluster 1	Cluster 2	x^2	p
Building Type	Care Services Buildings	10	0	25.741	0.001***
	Community Facilities	28	1		
	Core Offices	3	2		
	High Schools	18	3		
	Key and Cultural	4	2		
	Leisure and Sports	10	1		
	Parks Buildings	6	0		
	Primary Schools	63	0		
	Workshops and Depots	9	1		
Number of Floors	0	1	1	32.133	0.000***
	1	49	0		
	2	72	3		
	3	18	3		
	4	8	1		
	5	2	1		
	6	1	0		
	7	0	1		
Heating Type	Biomass	1	0	15.304	0.002***
	Electricity	0	1		
	Gas	9	149		
	Oil	0	1		
Energy Rating Certificate	A	2	0	12.56	0.084
	B	5	0		
	C	41	1		
	D	43	3		
	E	21	1		
	F	3	2		
	G	5	1		
	N	31	2		

$***p$ <0.01, $**p$ <0.05.

Table 3.13: Clusters profile summary for the four seasons.

Cluster	**Mean Load Usage**	**Max/min Load Usage**	**Cluster Label**
Spring			
1	4.39	7.87/2.42	Primary schools; Small floor numbers; Low load consumption; High energy efficiency.
2	24.82	39.18/14.38	Key and cultural; High floor numbers; Medium load consumption; Medium energy efficiency.
3	102.15	198.10/37.77	Core office; Medium floor numbers; High load consumption; Low energy efficiency.
Summer			
1	6.35	10.37/3.76	Primary schools; Small to medium floor numbers; Low load consumption; High energy efficiency.
2	79.84	149.48/37.44	Core office; High floor numbers; High load consumption; Medium to low energy efficiency.
Autumn			
1	6.73	12.42/3.31	Primary schools; Small to medium floor numbers; Low load consumption; Electricity/Gas/Oil heating.
2	64.46	103.72/35.13	Core office; High floor numbers; High load consumption; Biomass heating.
Winter			
1	8.21	14.52/4.32	Primary schools; Small to medium floor numbers; Low load consumption; Electricity/Gas/Oil heating.
2	75.00	120.21/42.86	Core office; High floor numbers; High load consumption; Biomass heating.

sumption are associated more with cluster 1, while cluster 2 has a higher load with a maximum load usage of 149.48 kWh in the summer. Moreover, electricity/gas/oil heating leads to lower load consumption in cluster 1 during the autumn and winter, whereas biomass heating leads to higher load consumption in cluster 2. During the warm season, i.e., the spring and summer, the load consumption is primarily determined by the building services and the number of floors, as summarized by the cluster label. Cold seasons, such as autumn and winter, are dominated by the load usage behaviour of the residents, which has a major influence on building load consumption.

3.4.3.3 Dynamic load usage behaviour recognition

After characterizing each cluster for the four seasons, a category summary is performed to track the dynamic cluster trajectories for the buildings. Table 3.14 summarizes all the cluster trajectories that appear in the buildings and lists the corresponding average load profile related to each trajectory in the four seasons, and the number of buildings corresponding to each DT.

Table 3.14: Category summary for the dynamic cluster trajectories over the year. DT: dynamic trajectory.

Spring cluster	**Summer cluster**	**Autumn cluster**	**Winter cluster**	**Spring Mean**	**Summer Mean**	**Autumn Mean**	**Winter Mean**	**Number of Buildings**	**DT No.**
1	1	1	1	4.39	3.74	4.56	5.49	139	1
2	1	1	1	19.05	20.18	24.89	27.64	17	2
		2	1	23.91	32.27	37.36	36.23	2	3
		2	2	30.51	31.78	42.85	49.52	3	4
	2	2	2	41.40	53.25	60.88	66.31	5	5
3	2	1	1	104.94	94.38	0.21	0.53	1	6
		2	2	100.75	139.03	132.93	134.96	2	7

A total of seven dynamic trajectories (DTs) appear in the clustering results for the 169 buildings. The dynamic load usage behaviour of each DT can be summarized by combining the cluster labels for each season defined in the previous section. The buildings in DT 1 have small to medium floor numbers and high energy efficiency, resulting in low load consumption. Compared with DT 1, DT 2-5 have relatively higher load consumption throughout the year. Particularly, DT 2 has a medium load consumption in the spring and decreases in load consumption

over the following three seasons. This may be because most buildings in DT 2 are heated by electricity, gas, or oil, which are energy-efficient and can reduce electricity consumption. The load consumption of DT 3 increases during the autumn, while the load consumption of DT 4 increases both during autumn and winter. In DT 4, most buildings are heated by biomass, which results in high load consumption during the winter. Furthermore, the DT 5 consumes moderate amounts of electricity during the spring, followed by higher electricity consumption from summer to winter. The smart grid operators should pay more attention to DT 3-5 during the winter season, as buildings in these DTs are more likely to have higher demand in the cold months. DT 5 should also be considered during the summer months as the residents of these buildings will consume more electricity during this period.

In spring and summer, most buildings following DT 6 have a low energy rating certificate. In contrast, in autumn and winter, the load consumption has decreased as most buildings are heated with electricity, gas, or oil. Buildings in DT 7, on the other hand, have high load consumption throughout the year. Most of these buildings have large floor areas and are heated by biomass during the cold months, which leads to high load consumption and low energy efficiency across the year. Smart grid operators should, therefore, ensure adequate load budgets for buildings in DT 6 during the summer months and buildings in DT 7 throughout the year.

3.4.3.4 Building load forecasting

The error indicators of mean squared error (MSE), MAE, mean absolute percentage error (MAPE), and R-squared (R^2) are considered in layer 4 to evaluate the forecast performance.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3.21}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{3.22}$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}|\frac{y_i - \hat{y}_i}{\hat{y}_i}| \times 100\% \tag{3.23}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(\bar{y}_i - y_i)^2} \tag{3.24}$$

where $\hat{y}_i$, y_i, and $\bar{y}_i$ are the observed i-th load consumption, the predicted electricity consumption, and the observed mean consumption, respectively. N denotes the size of the testing dataset, and i is the index of test observations.

Since MSE takes the square of the error, it will exaggerate the error caused by outliers. For MAE, it will be affected by the magnitude of the electricity consumption since it reflects the absolute error.

MAPE and R^2 can overcome the aforementioned limitations and are suitable for comparing different forecast algorithms for different buildings with difficult electricity consumption levels. Specifically, MAPE gives a percentage value, making the deviations comparable for different magnitudes of electricity consumption of different buildings. R^2, ranging from 0 to 1, is also a desirable evaluation indicator.

Since MAPE and R^2 give relative metrics to quantify the performance of different forecast models across different buildings, layer 4 only reports the scores of these two metrics.

To assist the forecasting process, load consumption-related features are selected, which are listed in Table 3.15.

Table 3.15: Selected features for hourly load forecasting

Input	Size	Description
L_h^{week}	1	h_{th} hour load on the same day of last week
L_h^{day}	1	h_{th}hour load of yesterday
L_h^{hour}	1	Load of the $(h-1)_{th}$ hour of today
F	2	One-hot code for festival/non-festival day
Y	2	One-hot code for year index
M	12	One-hot code for month index
W	7	One-hot code for day index of a week
H	24	One-hot code for hour index of a day

One-hot encoding is performed to represent festival/non-festival[1], year, month of the year, day of the week, and hour of the day. Based on the feature importance reported by RF, for each season, the first ten most important features are selected and displayed in Table 3.16-3.19, in which the first five most important features for each cluster were selected as the predictors to build the forecasting models.

Table 3.16: First 10 most important features in Spring

C1		**C2**		**C3**	
Importance	**Feature**	**Importance**	**Feature**	**Importance**	**Feature**
0.8904	last hour	0.9017	last hour	0.6992	last day
0.0395	last day	0.0450	last day	0.2258	last hour
0.0358	last week	0.0280	last week	0.0217	7:00
0.0024	9:00	0.0024	9:00	0.0158	last week
0.0024	0:00	0.0019	Monday	0.0061	20:00
0.0020	8:00	0.0014	8:00	0.0045	Tuesday
0.0017	Monday	0.0014	6:00	0.0037	Monday
0.0016	23:00	0.0013	7:00	0.0035	1:00
0.0016	22:00	0.0013	Saturday	0.0033	Wednesday
0.0015	Sunday	0.0010	10:00	0.0029	March

Key influential features vary between buildings, although the most critical feature (electricity consumption of the last hour) is the same for all the clusters, except for spring cluster 3, which takes the last day as the most important feature. In addition, the last day and last week are the two most common key features for all the clusters. For spring cluster 3, hour 7 are also important features. Combined with the cluster label in Table 3.13, most buildings in cluster 3 are the core offices, in which 7:00 may be the turning point for working/non-working hours. As a result, hour 7 forms an essential feature for those buildings.

In the spring months, Hour 9 is a key feature for clusters 1 and 2. Moreover, in spring cluster 1, most buildings are primary schools. Since most schools open on Monday, a sharp change in electricity use may occur. In other clusters with most primary schools, Monday has also been observed as an important feature. A key feature of winter cluster 1 is 23:00 rather than Monday, which might be because

[1]In the experiment, we considered the school holidays according to the historical school calendars of local schools in Cardiff, including four seasonal half-semesters, winter vacations, summer vacations, and bank holidays.

Table 3.17: First 10 most important features in Summer

C1		C2	
Importance	**Feature**	**Importance**	**Feature**
0.8999	last hour	0.7624	last hour
0.0584	last day	0.1297	last day
0.0251	last week	0.0552	last week
0.0011	Saturday	0.0133	6:00
0.0011	7:00	0.0085	7:00
0.0011	9:00	0.0069	Saturday
0.0010	Monday	0.0044	22:00
0.0010	8:00	0.0033	July
0.0007	0:00	0.0032	Thursday
0.0006	June	0.0028	festival

Table 3.18: First 10 most important features in Autumn

C1		C2	
Importance	**Feature**	**Importance**	**Feature**
0.8831	last hour	0.6390	last hour
0.0502	last day	0.2630	last week
0.0464	last week	0.0513	last day
0.0019	Monday	0.0053	Wednesday
0.0012	Saturday	0.0039	Saturday
0.0011	festival	0.0024	14:00
0.0010	9:00	0.0022	16:00
0.0009	19:00	0.0021	11:00
0.0009	18:00	0.0021	Tuesday
0.0009	8:00	0.0021	Friday

Table 3.19: First 10 most important features in Winter

C1		C2	
Importance	**Feature**	**Importance**	**Feature**
0.9124	last hour	0.5174	last hour
0.0350	last day	0.4033	last week
0.0306	last week	0.0396	last day
0.0016	23:00	0.0029	20:00
0.0014	9:00	0.0028	18:00
0.0013	Saturday	0.0020	15:00
0.0013	8:00	0.0019	December
0.0012	19:00	0.0018	Sunday
0.0012	10:00	0.0018	13:00
0.0011	January	0.0017	February

most of the buildings in cluster 1 are heated by electricity, gas, or oil. As a result, radiators/water tanks are heated and stored during off-peak hours (usually from midnight to seven in the morning), resulting in an increase in load consumption after 23:00. In comparison, the winter cluster 2, where most buildings are heated by biomass, will result in an increase in load consumption when the occupants use heating appliances in the cold night. As a result, 20:00 (a peak hour) has been observed as an important feature in winter cluster 2.

The first ten most important features for the aggregation level load data are listed in Table 3.20 to investigate the energy forecast performance based on the aggregation data of the 169 buildings. Compared with the feature importance at the individual building level, features that are unique to individual buildings (e.g., moths and festivals) are less critical for load forecasting at the aggregation level.

After selecting the five most important features for each cluster, forecasting models are established using SVR, RF, KNR, LSTM, and FR for each cluster, among which two models are built by SVR using linear kernel function and radial basis function (RBF), respectively. Therefore, there are 54 models for the load data from 169 buildings. We carry out a grid search for parameters tuning, and the implementation steps are as follows:

1. Give a range of possible parameters for the five models; their parameter ranges

Table 3.20: First 10 most important features for the 169 aggregated buildings

Importance	**Feature**
0.8692	last hour
0.0560	last week
0.0416	last day
0.0080	7:00
0.0078	8:00
0.0024	9:00
0.0022	Monday
0.0015	10:00
0.0012	21:00
0.0010	0:00

are determined via a trial and error manner. The range of parameters is shown in Table 3.21.

Table 3.21: Optimal parameters for each forecast model

Model	Parameters and range
SVR(Linear)	C:[0.01, 0.1, 1, 10]
SVR(RBF)	C:[0.01, 0.1, 1, 10], γ:[0.001, 0.01, 0.1, 1]
RF	max_depth:[5, 10, 15, 20], min_sample_split:[2, 5, 10, 15], n_estimators:[10, 100]
KNR	k-neighbours:[1-10]
LSTM	activation:[‘relu’, ‘tanh’, ‘sigmoid’], optimizer:[‘SGD’, ‘Adam’], epochs:[10, 15, 50, 100, 150], batch size:[16, 32, 64, 128]

2. Parameters are paired to form parameter grids.

3. Relevant parameters are successively substituted into the model for the network nodes, and the optimal parameter combination is selected according to the best results.

Based on the above steps, the best parameter combinations for models at the individual and aggregation levels are listed in Table 3.22.

Table 3.22: Best Parameters for each model

Model	**Parameter**	**Spring**			**Summer**		**Autumn**		**Winter**		**Aggregated**
		C1	**C2**	**C3**	**C1**	**C2**	**C1**	**C2**	**C1**	**C2**	**Data**
SVR(Linear)	C	0.01	10	0.01	0.01	0.01	0.01	0.01	0.01	10	0.1
SVR(RBF)	C	10	10	10	10	10	10	10	10	10	10
	γ	0.001	0.01	0.001	0.001	0.001	0.001	0.001	0.001	0.01	0.01
RF	max_depth	15	15	10	15	10	15	15	15	10	15
	min_sample_split	10	10	2	15	10	15	15	15	10	5
	n_estimators	100	100	10	100	100	100	100	100	100	100
KNR	k-neighbours	5	2	2	5	2	6	2	5	2	2
LSTM	activation	'relu'	'relu'	'relu'	'relu'	'relu'	'relu'	'relu'	'relu'	'relu'	'relu'
	optimizer	'Adam'	'Adam'	'Adam'	'Adam'	'Adam'	'Adam'	'Adam'	'Adam'	'Adam'	'Adam'
	epochs	100	150	100	50	50	15	100	15	10	150
	batch size	32	32	32	128	32	128	128	16	64	16

Models were built for each cluster of each season based on the selected forecasting algorithms, and total 45 models has been built at the building level. The forecast accuracy at the individual building level are reported in Table 3.23. Besides, load data at the aggregation level of different models are also reported in this table.

Overall, the aggregation level forecasting model performs better than the individual level forecasting model with the lowest MAPE of 2.48 %, and the highest R^2 of 0.97. The LSTM performed best at the individual building level from summer to winter, whereas it performed less well in autumn cluster 2 and winter cluster 2. However, the FR achieved the lowest MAPE for these two clusters. Cluster 2 likely has fewer buildings during autumn and winter, which prevents the LSTM model from learning sufficient features to predict load usage patterns correctly. FR, however, can effectively extrapolate the shape of predicted load curves by considering the discrete load samples as individual load curves.

It is worth mentioning that the forecast scores for summer cluster 2 and spring cluster 3 are worse than other clusters; most forecasted R^2 are below 0.91. An intuitive explanation is that the data distribution of these two clusters is very different

Table 3.23: Forecasting results for each model

Model	Evaluation	Spring			Summer		Autumn		Winter		Aggregated
	Indicators	C1	C2	C3	C1	C2	C1	C2	C1	C2	Data
SVR(Linear)	*MAPE* (%)	8.52	8.40	19.43	9.22	7.41	10.26	6.75	8.36	6.18	5.59
	R^2	0.89	0.91	0.79	0.94	0.82	0.92	0.84	0.94	0.83	0.94
SVR(RBF)	*MAPE* (%)	9.27	7.91	17.44	8.69	7.24	9.12	6.01	8.24	5.53	3.68
	R^2	0.91	0.94	0.87	0.96	0.83	0.95	0.83	0.96	0.89	0.91
RF	*MAPE* (%)	8.59	5.10	6.81	8.19	**5.51**	7.26	4.81	8.46	5.17	3.00
	R^2	0.93	0.95	0.92	0.97	**0.97**	0.96	0.88	0.96	0.87	0.96
KNR	*MAPE* (%)	8.39	**4.47**	**5.60**	6.90	6.46	7.63	4.96	7.95	5.29	2.83
	R^2	0.93	**0.95**	**0.96**	0.96	0.90	0.94	0.88	0.96	0.85	0.93
LSTM	*MAPE* (%)	8.13	9.41	8.68	**5.11**	6.87	**6.22**	6.55	**6.55**	7.02	**2.48**
	R^2	0.94	0.94	0.91	**0.96**	0.82	**0.97**	0.85	**0.97**	0.84	**0.97**
FR	*MAPE* (%)	**7.45**	9.42	10.11	6.49	11.27	6.33	**4.74**	7.58	**2.63**	3.14
	R^2	**0.94**	0.93	0.88	0.95	0.79	0.96	**0.90**	0.95	**0.95**	0.95

from the other clusters, as can be seen from the former Figure 3.19; the cluster centroids of these two clusters are much different from other clusters, which have more load peaks and valleys. Therefore, it can be observed that models including SVR, LSTM, and FR perform relatively worse than the RF and KNR. Notably, the FR has a relatively worse performance with the MAPE higher than 10%, indicating that the functional representation may smooth some useful features for forecasting. On the other hand, RF and KNR achieved better scores for both clusters, and it may be because the RF and KNR are two algorithms that are not sensitive to the outliers, which can train robust models even if there are more fluctuations in the load curves.

Overall, LSTM and FR models outperform other models in most clusters, followed by SVR models. Since RF and KNR are not sensitive to outliers, they are suitable for training robust models for load data with more fluctuations. Moreover, based on the comparison of the forecast performance with the individual buildings, aggregation can mitigate the high randomness in electricity consumption of individual buildings.

3.4.3.5 Discussion

We can conclude that there are different physical factors contributing to load usage behaviour differences among clusters in different seasons. Furthermore, the dynamic cluster trajectory of the buildings provides valuable guidance for designing statistic and dynamic load pricing strategies.

Moreover, the above promising results can also be applied to other relevant applications such as for the building thermal management. Based on the customized data-driven building load profiles, customized electricity management approaches can be developed for each group of buildings. It is also possible to consider customized energy pricing and demand response management strategies for managing the energy costs. Based on the above, the proposed system has promising computational benefits. The consensus-based clustering and training method improves model development efficiency significantly, making it useful for large regions with multiple building types, as well as big data scenarios.

3.5 Chapter Summary

In this chapter, firstly, we introduce an ADLs recognition NILM model to detect and infer the daily activities of smart meter users based on the appliance level load analysis. More specifically, we propose an improved deep neural network model based on sequence-to-point and transfer learning, which aims to optimize the training efficiency of the model while ensuring the accuracy of the load disaggregation results. Moreover, we utilize useful data analysis tools such as PCA and k-means based on the disaggregated appliance consumption to conduct ADLs pattern recognition for smart meter users. The comparative experimental results show that our improved model can efficiently disaggregate the usages of target appliances in the unseen house, and the usage regularities of target devices can be effectively inferred.

After exploring the appliance-level load analysis, this chapter proposes a consensus-based load profiling and forecasting system to conduct individual and higher level load analysis, in which the consensus-based robust clustering approach is applied to cluster individual buildings into different groups, to capture different building load usage behaviours in different seasons. By analysing clustering results with accompanying building physical information, we can identify underlying factors that explain the observed load usage behaviours. This is an important step for the following forecasting layer to develop personalized building load forecasting models.

Chapter 4

From Offline to Online: Real-time Smart Meter Data Analysis

4.1 Introduction

Analysing smart meter data is imperative to balance energy consumption and minimize power outages. Traditional load forecasting techniques rely on historical consumption patterns to obtain load forecasts. However, such techniques are difficult to adapt to dynamic changes in newly arrived real-time load curves. Moreover, most studies analysing residential smart meters focus on understanding the load records based on individual participants rather than analysing the load records, which are unlikely to be generalized to other forecasting levels. To address these challenges, this chapter proposes a two-unit universal online functional analysis model (Universal-OFA) with universal applicability for dynamically profiling and forecasting multi-scale demand. With respect to Chapter 3, this chapter considers a more complex and practical scenario, i.e., the real-time smart data analysis, which is adapted from our two papers 5 and 7.

The rest of this chapter is organized as follows. Firstly, the problem statement is given in Section 4.2. Then, Section 4.3 describes the detailed methodological approach for the Universal-OFA and discusses its benefits. Specifically, the overview of the proposed framework is given in Section 4.3.1, and the implementation detail is then described in Section 4.3.2. Thirdly, Section 4.3.3 presents the data used in the experiments and conducts comparative experiments for evaluating the two units in Universal-OFA based on real-world data. Section 4.4 gives a conclusion to this

chapter.

4.2 Problem Statement

With the development of smart grid technologies in recent years, electricity consumption can be monitored in finer detail through the installation of AMI. AMI consists of smart meters capable of two-way communication that transmit information between utility companies and the consumers [93]. The energy system can achieve mutually beneficial outcomes between energy supply and demand by facilitating demand side management and minimizing power outages [42]. Residential buildings make up a significant proportion of end-use electricity demand [203]. Amongst the forecast studies targeting the residential sector, the main focus has been on aggregate region or district level electricity loads that aggregate all involved participants and, therefore, generate a smoothing effect to improve the accuracy of time series forecasting [62].

However, the behaviour of resident groups under the same electricity pricing strategy may differ greatly in a region due to different social/economic backgrounds. Price-sensitive residents may reduce their load consumption to avoid paying higher electricity bills during periods of high electricity prices. Price-insensitive residents may be more concerned with their living comfort than with additional expenses and are less sensitive to changes in electricity tariffs [64]. Consequently, it is important to understand the load profile and its associated social characteristics of residents rather than simply aggregating their load usage.

On the other hand, existing load forecasting models at the individual level overemphasise analysing electricity patterns by individual households. Developing load forecasts for individual households is challenging because of the high variability caused by the dynamic processes that consist of many uncertain and random load curves. Focusing on the individual load curves rather than households might solve the above challenge.

Furthermore, individual households may enter into the smart meter analysis model asynchronously, which results in the loss of effectiveness for many previously trained models and requires continuous retraining to adapt to dynamically changing smart grid environments. Moreover, existing residents may contain various social/economic information that corresponds to various specific electricity consumption patterns. As new residents join, more types of social information will be incorporated into individual-level profiles, resulting in a richer contextual social/economic profile for the specific electricity consumption patterns.

Therefore, we proposed the Universal-OFA, which assumes that only a por-

tion of the participants in a region can provide historical load and that the future individual load of these participants and any other newly joined participants, as well as the load of other higher aggregation levels, will be predicted with a loose sketch-detail refinement approach. Moreover, to assist in defining future personalized electricity pricing strategies, this chapter also examines the relationships between social/economic background and individual level load usage patterns, as well as the dynamic changes in household load usage behaviours.

4.3 A Universal Online Functional Analysis Model for Multi-scale Load Dynamic Profiling and Forecasting

4.3.1 Framework Design Overview

In the Universal-OFA model, each load sequence is composed of discrete time points sampled from a finite equidistant mesh and treated as random function drawn from a continuous stochastic process $X = \{X(t); t \in \mathbb{R}\}$, where t is a time point on the discrete time grid. In the model, we assume that we have observed historical load values for the process X over an interval $[0, T]$ from only a portion of the participants in a region, and our goal is to forecast the load values of X over a future interval $[T, T + \sigma](\sigma > 0)$ at different levels, including the individual level (existing or new) participants and the region level aggregated load usage. The individual level historical interval $[0, T]$ can be divided into N sub-intervals with a pre-segmented length β. Denoting by $L_{n_d}(t_i)$ the functional-valued discrete stochastic process over the sub-interval n and day d, we have

$$L_{n_d}(t_i) = X_n((d-1)\beta + t_i), \tag{4.1}$$

where $X_n(t)$ is the restricted real-valued continuous stochastic process of $X(t)$ on the n-th interval, and is translated on the interval $[0, N \cdot \beta]$. The sub-interval $n \in \{1, 2, ..., N\}$, and $i \in \{1, 2, ..., P\}$. In our case, we use each load curve observed from half-hourly load records of a day d to forecast the half-hourly load values of the next day $d + 1$. Therefore, the assigned length $\beta = \sigma = P = 48$. For the sub-interval n on day d, we have

$$L_{n_d} = [L_{n_d}(t_1), L_{n_d}(t_2), ..., L_{n_d}(t_{48})], \tag{4.2}$$

and the Universal-OFA is to forecast the load values of the sub-interval n on the day $d + 1$ at different levels:

$$L_{n_{d+1}} = [L_{n_{d+1}}(t_1), L_{n_{d+1}}(t_2), ..., L_{n_{d+1}}(t_{48})]. \tag{4.3}$$

where $\mathbf{t} = (t_1, \cdots, t_{48}) \in \mathbb{R}$. For each $1 \leq p \leq 48$, the random function $X(t_p)$ is square-integrable with the inner product of any two real-valued functions f and g defined as

$$< f, g >_{\mathcal{T}_p} = \int_{\mathcal{T}_p} f(t_p) g(t_p) dt_p, \qquad f, g \in \mathcal{H} \tag{4.4}$$

where $\mathcal{H} := \mathcal{L}^2(\mathcal{T}_1) \times \cdots \times \mathcal{L}^2(\mathcal{T}_{48})$ is a Hilbert space, which is a vector space that enables the expansion of linear algebra and calculus from finite-dimensional Euclidean vector spaces to spaces that could be infinite-dimensional, and $\mathcal{T} := \mathcal{T}_1 \times \mathcal{T}_2 \times \cdots \times \mathcal{T}_{48}$. The inner product $< \cdot, \cdot >$ induces the norm $\| f \| = \sqrt{< f, f >_{\mathcal{T}_{48}}}$. Let μ denote the smooth mean function of X, and C denote the covariance function of X, we have

$$\mu(\mathbf{t}) = \mathbb{E}(X(\mathbf{t})), \tag{4.5}$$

$$C(\mathbf{s}, \mathbf{t}) = \text{cov}(X(\mathbf{s}), X(\mathbf{t})). \tag{4.6}$$

The structure of the Universal-OFA model is shown in Figure 4.1, which is designed with two units, i.e., the multi-scale load dynamic profiling unit, and the multi-scale online load forecasting unit. Each unit consists of two layers that follow a loose sketch-detail refinement strategy.

Based on the historical load records of the individual level participants, the loose sketch strategy at layer one of Universal-OFA enables the development of universal models for the two units. The loose features of the intra-day load usage patterns and their associated social information will be sketched, and universal FDN models will be trained at this layer. The detail refinement strategy in layer 2 for the two units in Universal-OFA allows the universally sketched models to consider the time-varying load consumption patterns for the detailed refinement of the universally-sketched clusters and universally trained FDN models at different levels.

Initially, the multi-scale load dynamic profiling unit conducts functional universal-sketching to cluster the historical load records of individual level participants. In this layer, an intra-day volatility score was proposed to assist the clustering process. Multi-scale dynamic profiling will then be performed using the real-time load sequences collected from individual participants (existing/new) and regional levels to track the dynamic change at different levels over time. The multi-scale load dynamic profiling unit combines load records with social backgrounds to examine the relationship between different types of participants and the corresponding load usage behaviours.

The multi-scale online load forecasting unit will make use of universally-sketched clusters to perform functional deep neural network universal training on the loose

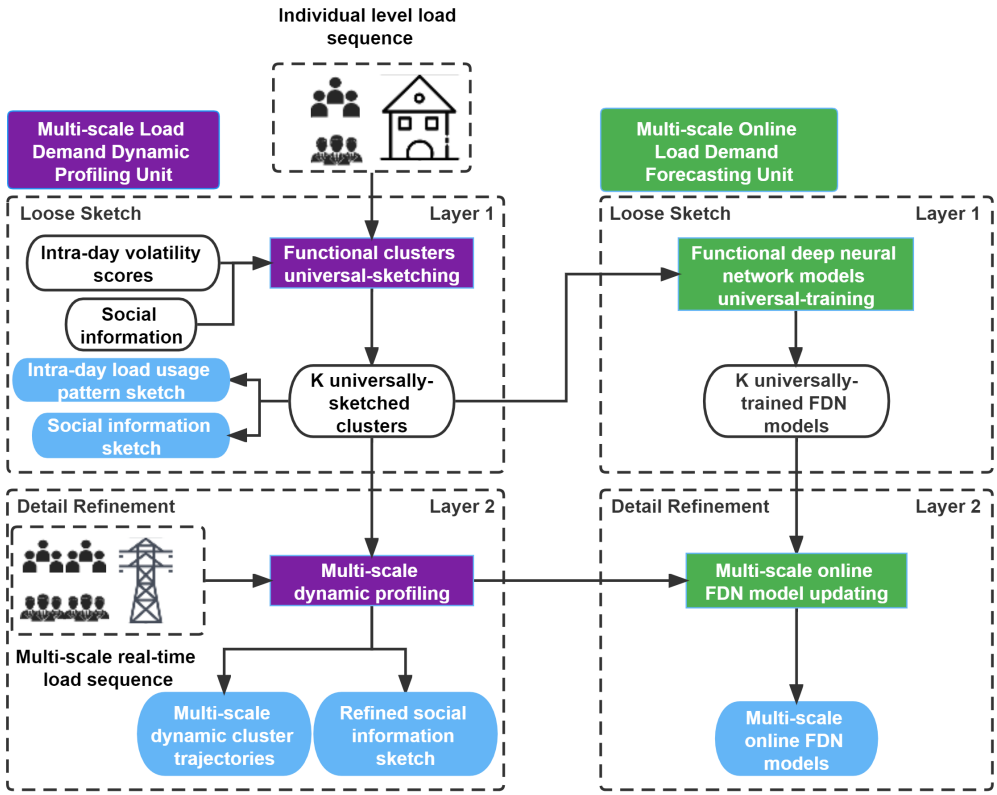


Figure 4.1: The proposed Universal-OFA model.

sketch layer. These universally-trained models have been trained to capture features of the historical load usage patterns of existing participants. By exploiting the dynamically profiled real-time load sequences at different levels, these universally-trained models will be incorporated into the multi-scale FDN model updating process at layer 2. Upon passing through layer 2, personalized online FDN models will be developed for each cluster at different levels.

4.3.2 Implementation

4.3.2.1 Multi-scale load dynamic profiling unit

To assist in the development of universally-sketched models, the functional historical load records of the individual participants are stratified into multiple groups with similar load usage patterns. In this chapter, we are using a different approach than in Chapter 3. We are utilizing model-based functional data clustering which views the original daily load discrete points as continuous load curves. Instead of categorizing the data by season like in Chapter 3, the focus of this method is to identify load curves with similar shapes by analyzing the load consumption features of different hours in a day. This allows for a more accurate capture of daily fluctuations in the load curves. Many studies have demonstrated that selecting the appropriate attributes for load analysis is critical [76]. In this study, the primary objective is to identify more features of the intra-day load patterns to assist the functional data clustering process. So we investigate the impact of the Time-of-Use tariff [14] to further improve the clustering process, which is shown in Table 4.1.

Table 4.1: The Time-of-Use tariff.

Period	**Time**	**Price**
Daytime	08:00 to 16:30 and 19:00 to 23:00	Medium
Peak	17:00 to 18:30	High
Overnight	23:00 to 07:30	Low

According to the Time-of-Use tariff, there are three time periods: daytime, peak, and overnight. Prices of electricity are highest during peak hours and lowest at night. The three time periods in the Time-of-Use tariff are indexed by 1, 2, and 3, corresponding to the daytime, peak, and overnight periods, respectively. These periods are then assigned an intra-day volatility score (IVS):

$$IVS_i = \frac{E_i}{\bar{L}}, \tag{4.7}$$

where E_i is the mean half-hourly load at period i, and $\bar{L}$ calculates the average daily load consumption. Considering that there may be a particular period of non-use of electricity by consumers, a minimal positive value of 10^{-6} was added to the denominator of each mean calculation to avoid the denominator being zero. IVS measures the fluctuation of load within a specific period and is useful for determining the optimal demand management strategy and assessing the volatility of intra-day load.

After finding the IVSs for the load patterns, a model-based functional data clustering procedure [72] will be used to conduct unsupervised clustering for the daily load curves, which eliminates the need to determine the cluster number. Recall the functional-valued discrete stochastic process $L(t)$ defined in Eq. (4.1), the model based clustering assumes $L(t)$ follows a functional mixture model with K subprocesses, with each subprocess corresponding to a cluster k , and a Karahunen-Loève representation for daily load consumption of L allows decomposition:

$$L(t) = \sum_{k=1}^{K} \mu_k(t) + \sum_{j=1}^{\infty} \xi_j \phi_j(t) \tag{4.8}$$

where μ_k is the smooth mean curve function in Eq. (4.5) in the k-th cluster, and ξ_j is the real-valued variable with an orthonormal basis ϕ_j, j is a real-valued random variable of $\mathcal{H}$, and $\mathcal{H} := \mathcal{L}^2(\mathcal{T}_1) \times \cdots \times \mathcal{L}^2(\mathcal{T}_P)$ is a Hilbert space.

The representative scores of the original features including the IVSs and the load data then need to be found to construct an unsupervised binary tree considering the mixture model built in Eq. (4.8). Since the IVSs and the load data are defined on potentially different domains, multivariate functional principal components analysis (MFPCA) [79] is employed to represent the functional data by computing the eigenfunctions for each domain separately. Specifically, the local polynomial smoothing with Gaussian kernel [189] is utilized to estimate the mean (Eq. 4.5) and covariance (Eq. 4.6) of the original dataset on each domain. A fitted smoothed example of the load records compared with the true values is shown in Figure 4.2.

The eigenfunctions $\Phi_{\gamma,j} = (\phi^1_{\gamma,j}, \cdots, \phi^C_{\gamma,j})$ and eigenvectors $\Lambda_{\gamma,j} = (\lambda^1_{\gamma,j}, \cdots, \lambda^C_{\gamma,j})$ are computed as a matrix eigenanalysis of the covariance. Then for each domain $\theta \in \{1, \cdots, \Theta\}$, the matrix of scores $S^\theta_{\gamma,j}$ representing the projection with C components of the load curves of $L(t_i)$ onto $\Phi_{\gamma,j}$ are estimated by numerical integration. Finally, the multivariate scores $S^\Theta_{\gamma,j}$ are estimated by plugging the computed scores of each domain.

After finding the representative scores of the original data, a full binary tree with a root node $\Gamma_{0,0}$ is constructed. The tree has two types of nodes, the non-terminal nodes with two disjoint children subsets from $L(t)$ and the terminal nodes with no

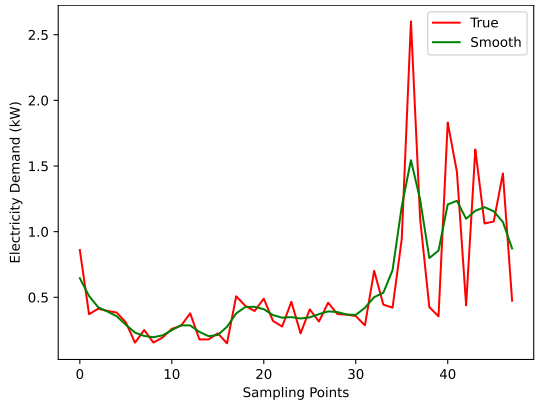


Figure 4.2: An example of the true daily load readings compared with the smoothed functional curves.

children. The nodes are indexed by (γ, j), where γ is the depth index, and j is the node number index. Each node $\Gamma_{(\gamma,j)}$ is possibly to have a left child $\Gamma_{(\gamma+1,2j)}$ and a right child $\Gamma_{(\gamma+1,2j+1)}$.

To avoid over-partition, a threshold K_{max} is adopted. At each node $\Gamma_{(\gamma,j)}$, for each $K = 1, \cdots, K_{max}$, the scores $S^{\Theta}_{\gamma,j}$ are fitted with the Gaussian mixture model (GMM). Then the Bayesian information criterion (BIC) [186] is used to estimate the optimal number of clusters $\widehat{K}_{\gamma,j}$:

$$\begin{aligned} \widehat{K}_{\gamma,j} &= \underset{K=1,\cdots,K_{max}}{\arg\max} \ \mathrm{BIC}(\mathcal{M}_1, \cdots, \mathcal{M}_K) \\ &= \underset{K=1,\cdots,K_{max}}{\arg\max} \ \{K \ln |\Gamma_{\gamma,j}| - 2\ln(\mathcal{L}_K)\}, \end{aligned} \tag{4.9}$$

where $\mathcal{M}$ is the fitted GMM model, $\mathcal{L}_K$ is the likelihood of K subprocesses $\mathcal{M}_K$ for the node $\Gamma_{\gamma,j}$. If $\widehat{K}_{\gamma,j} > 1$, $\Gamma_{\gamma,j}$ will be divided into two children nodes. Otherwise, $\Gamma_{\gamma,j}$ is a terminal node. The recursive process is repeated until it satisfies the stopping rules, and then a joining step is performed to rectify the redundant nodes. The functional clusters universal-sketching based on the historical functional load consumption data is completed at this point. Afterward, social information related to each cluster will be summarized to produce more interpretable results. The intra-day load usage pattern sketch and the social information sketch will be generated at this layer, which will be refined in more detail at the subsequent layer.

A dynamic clustering algorithm is applied to layer 2 to cluster the newly arrived daily load records based on the tree constructed in the functional clusters universal-sketching layer. Based on the projection of the newly received sequence

onto eigenfunctions for each node of the tree, the posterior probability of this sequence belonging to each component will be estimated again using the GMM. The new sequence is assigned to the cluster candidate with the highest likelihood. Using the loosely sketched outputs from layer 1, the multi-scale dynamic profiling module will refine social information and identify the dynamic cluster trajectories of each participant at different time scales.

4.3.2.2 Multi-scale online load forecasting unit

In the multi-scale online load forecasting unit, functional deep neural network (FDN) models were trained based on universally-sketched clusters. The proposed FDN is shown in Figure 4.3, which utilizes a micro basis layer proposed in [222]. The basis layer [222] parameterizes the representation functions with a micro neural network and uses numerical integration to weigh and calculate the final score to approximate the true inner product between the basis functions and the input values. Unlike the previous work in [222], which predicted a single value at a one-time step, the goal of the FDN models universal-training layer is to train universal FDN models that forecast half-hourly load daily and capture loose features of historical load usage patterns at the individual level.

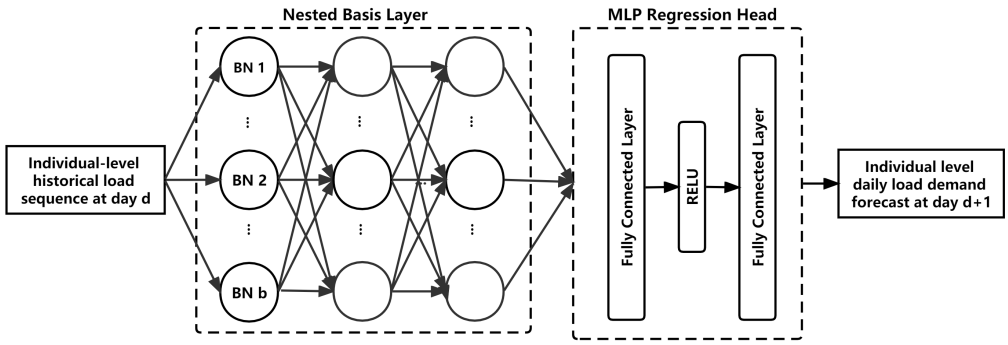


Figure 4.3: The architecture of the proposed FDN.

Recall that $L_{n_d}(t)$ is the discrete stochastic process on the sub-interval n and day d. Let $\{\phi_j(t)\}_{j=1}^{b}$ be a set of b continuous basis functions. The output of the j-th basis node is a collection of score vectors for the basis functions:

$$c_j =< \phi_j, L_{n_d} >= \int \phi_j(t) L_{n_d}(t) dt. \tag{4.10}$$

In practice, each basis function is parametrized with a micro neural network at each time step, and the integral in the above formula needs to be approximated numerically. The nested basis layer consists of 3 hidden layers and 128 fully connected nodes per layer. The score vectors are then input into the multilayer perceptron (MLP) regression head, and the divergence between the forecasted value and the actual response is evaluated to determine the loss of the network.

Although various functional data clustering and forecasting methods have been proposed, most existing works rely solely on historical functional inputs, which makes trained models incapable of adapting to newly arrived load curves over time. In the second layer of Universal-OFA, a new online updating strategy is proposed by leveraging the information feedback from real-time smart meter readings to adaptively update and optimize the universally-trained FDN models in layer 1. As new load records are generated by existing participants or as new participants join the system, the multi-scale online FDN model updating module forecasts the load values of X at different levels over a future interval $[T, T + 48]$.

A majority of existing functional data forecasting studies have focused primarily on training and testing forecasting models without considering dynamic changes in electricity consumption behaviour. By incorporating feedback from newly arriving smart meter readings at different levels, the online multi-scale FDN model updating module continuously refines the details of universally trained models, allowing personalized forecasting models to be developed through online learning.

The online updating strategy of the multi-scale online FDN model updating module is shown in Figure 4.4. For the k-th model concerning the k-th cluster, the MLP regression head is updated by online learning (updating batch size=1) when a new load sequence is received. Specifically, in each FDN model, recall that the goal of the framework is to forecast $L_{n_{d+1}}$, given the newly observed daily load records L_{n_d} for the sub-interval n on the day d. We first use L_{n_d} to forecast the predicted value $\hat{L}_{n_{d+1}}$, and then use it along with the true value $L_{n_{d+1}}$ revealed on the next day $d+1$ as the input to the regression head to optimize the model parameters. As a result, the basis layer will remain unchanged since it extracts the functional features of historical inputs, and only the MLP layers will be updated based on the received load sequences to guide the model in forecasting the daily load more accurately. The frozen step of the basis layer ensures that the online learning process does not deviate too much from the correct gradient direction while also providing the optimization process with the flexibility to capture more dynamic changes in the time-varying load.

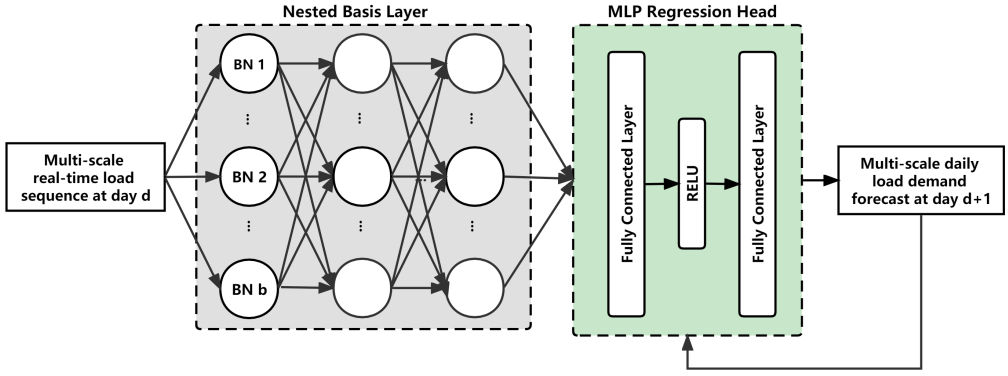


Figure 4.4: Multi-scale online FDN model updating process.

4.3.2.3 Discussion

There are several practical considerations associated with the proposed Universal-OFA model. In a particular region, individuals with different social backgrounds have different electricity consumption patterns and levels of sensitivity to electricity prices. Most studies cluster load usage patterns based on individual IDs, i.e., different clusters contain different participants. Consequently, if load usage patterns are derived from their IDs without preserving personal information, the daily routine habits of participants may be revealed. Smart meter analysis may be limited to a small number of participants in real-world scenarios due to privacy concerns. Therefore, traditional smart meter analysis models can only be applied to a limited number of participants and cannot be applied universally to new participants due to a lack of generalisation capability.

Unlike traditional frameworks, we emphasize that clustering and training of the loose sketch layer (layer 1) in both Universal-OFA units are ID-free processes, in which historical load records are collected anonymously into a common pool for undifferentiated clustering and training, such that load curves over several consecutive days of the same participant can be categorized into different groups. By doing so, additional flexibility can be provided in analysing load usage patterns and ensuring the participants' anonymity, i.e., their identity remains anonymous.

Both units use the loose sketch layer for clustering and training the models, which captures the patterns of load usage based on daily load curves rather than all the load records of each participant. Considering the variable and uncertain nature

of residential energy consumption, a model trained on a participant-by-participant basis cannot forecast accurately when a random load curve occurs since the random load curve contains new features that have not been observed previously. A model trained on electricity usage patterns can, on the other hand, cluster and predict based on the features of the load patterns, which ensures accuracy in the individual level load forecasting.

We emphasize that the individual level analysis and the regional level analysis in the experiment are only two illustrative examples that demonstrate the generalizability of the proposed Universal-OFA. The detail refinement layer in the two units can analyze real-time load at different levels, such as the individual, the city, the region, and the country. Furthermore, both sketches in layer 1 of the two units are universal models that can be generalized to a broader range of data by learning supplementary features from real-time load records at various data scales.

With the multi-scale dynamic profiling, social information from newly joined participants is considered in conjunction with the previous profiles, resulting in an overall more comprehensive contextual social/economic profile of electricity consumption patterns for each cluster, which may assist in deriving electricity consumption characteristics for participants with different social backgrounds. Further, the multi-scale dynamic profiling facilitates the tracking of dynamic changes in the electricity usage behaviours of each participant, which enables quantification of the effectiveness of various dynamic-based and incentive-based demand response strategies.

In real-world scenarios, where participants may join the smart meter analysis system asynchronously, it is difficult for traditional models to train accurate models for new participants. Nevertheless, the proposed model requires universal training of historical load usage patterns from only a portion of participants, making it more efficient when new participants join asynchronously. Rather than starting from scratch, the Universal-OFA trains personalized models for the existing/new participants in the multi-scale online FDN model updating module by adding their detailed load usage features to the previous universally-trained models. Using real-time smart meter data at different data scales, the proposed Universal-OFA can learn to adapt to the newly arriving load sequence by incorporating time-varying load consumption patterns.

4.3.3 Evaluation and Discussion

4.3.3.1 Data description and experiment settings

Residential load records from the Commission for Energy Regulation (CER) on half-hourly domestic electricity consumption in Ireland [14] for a whole year from July 14, 2009, to July 14, 2010, were utilized. 3440 residents remain after the missing values have been removed from the datasets. The dataset includes a pre-trail survey, which provides information about the socioeconomic background of each participant, allowing us to interpret the clustering results more clearly.

As the number of rooms in a household is directly related to the amount of electricity consumed, we selected participants according to the proportion of different numbers of rooms in the total number of residents, as shown in Table 4.2. A weighted random selection was conducted to select half of the residents as individual level participants and to select the remainder as new participants who would join subsequently. 1717 residents were randomly selected as the existing participants, and the remaining 1723 residents were randomly selected as new participants.

Table 4.2: The number of selected participants.

No. Rooms	Proportion	No. Selected Participants
1	1.0%	17
2	8.4%	145
3	44.3%	762
4	34.9%	600
5	11.2%	197

There are 365 days per participant. We select the first 265 days (from July 14, 2009, to April 06, 2010) as the historical load records and the remaining 100 days (from April 07, 2010, to July 14, 2010) as the online smart meter readings. In the case of new participants, only 100 days (from April 07, 2010, to July 14, 2010) are used as online smart meter readings since we assume their historical load records still need to be provided. Moreover, to evaluate the universal applicability of the Universal-OFA model, we use the aggregated mean of 100 days (from April 07, 2010, to July 14, 2010) of all the 3440 participants as the region level online smart meter readings. This partitioning is adopted to ensure that the framework can access sufficient data to perform reliable clustering and model training. Moreover,

it allows us to observe the impact of this framework on adaptive model updating as new unseen smart meter readings are acquired over time.

It is worth noting that this forecasting task considered in this study is more complex than many previous works [31, 139, 71]. For training, it utilizes the entire autumn and winter seasons, the last two months of summer (July and August), and the first two months of spring (March and April), while for testing, the last two months of spring (April, May) and the first two months of summer (June, July) are used, where May and June are not observed by the trained model and may contain new patterns in electricity consumption.

Based on the above partitioning, the goal of the Universal-OFA is to use the observed discrete stochastic process $L_{n_1}, L_{n_1}, ..., L_{n_{265}}$ with each day have half-hourly smart meter readings of the individual level participants to forecast $L_{n_{266}}, L_{n_{267}}, ..., L_{n_{365}}$ for the newly joined participants and the newly generated load sequence at the individual and the region level. The experiments were implemented using Pytorch and conducted on a Windows 10 platform (64GB RAM) with GPU NVIDIA GeForce RTX 2080 Ti and CUDA v10.2.

4.3.3.2 Multi-scale load dynamic profiling results

- **Interest of the intra-day volatility scores**

  This subsection aims to evaluate the validity of the IVSs by comparing the BIC of the results obtained with and without the IVSs, as shown in Table 4.3.

Table 4.3: BIC scores

	BIC
Without IVSs	2.05×10^6
With IVSs	3.23×10^6

  As described in Eq. (4.9), a higher BIC indicates better clustering results, and Table 4.3 shows that the cluster model with IVSs has a higher BIC than the cluster model without IVSs, demonstrating that IVSs can improve the clustering process.

- **Functional Clusters Universal-Sketching Results**

  During the clustering, 99% of the components were preselected in MFPCA, and a sample size of 55,000 was selected to avoid over-partitioning. After applying the functional clusters universal-sketching module, 7 clusters were obtained from total N = 265*1717 individual-level load curves, which are shown in Figure 4.5.

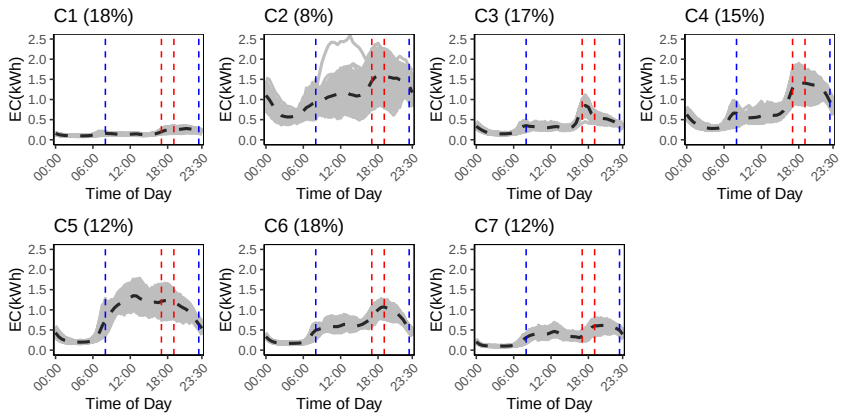


Figure 4.5: Functional clusters universal-sketching results. The red dashed vertical lines denote the peak period (17:00–18:30), the blue dashed vertical lines denote the daytime period (8:00–16:30 and 19:00–23:00), and the black dashed vertical lines denote the overnight period (from 23:00 to 07:30).

Further, the class of social background with the largest share in each cluster is summarized as the representative social information for each cluster, which will be combined with average daily load and IVSs to assign a loose sketch label to each universally-sketched cluster. In Table 4.4, we summarized the above information and marked the average daily load and the IVS in each period with arrows to indicate high or low electricity consumption in accordance with the median.

Clusters 1 and 6 appear to have the most records with 18%, followed by clusters 3 with 17%, cluster 4 with 15%, and clusters 5 and 7 with 12%. Less than 10% of total load records are contained in cluster 2. Clusters 2, 4, and 5 describe load usage behaviour with relatively higher load consumption, and we can see that the average daily load of cluster 2 is the highest compared with other clusters. Across all three clusters, there are 5 rooms and 5-6 residents, which indicates that having more rooms and residents will result in greater electricity consumption.

Furthermore, clusters 2 and 4 have higher social classes and income levels than cluster 5. Despite cluster 2 having the highest average daily load consumption, IVSs for all three periods remain stable. Moreover, cluster 4 has a high IVS during peak periods and a low IVS during overnight periods despite peak periods having a higher electricity price than overnight periods.

Although cluster 5 has a higher average daily load consumption, the IVS during peak time indicates that residents in cluster 5 have responded to the change in electricity price. Based on average daily load consumption and IVSs for clusters 2 and 4, these cluster participants are less price sensitive. They are more concerned with living comfort without considering the extra expenses caused by Time-of-Use tariffs.

Table 4.4: The social information sketch and the IVSs of the universally-sketched clusters.

	No. adults	**Social Class**	**No. rooms**	**Income Level**	**Avg. Daily Demand**	**Daytime**	**Peak**	**Overnight**
C1	1	F	1	1	0.16(↓)	1.08(↓)	1.24(↓)	1.11(↓)
C2	5	AB	5	5	1.09(↑)	1.06(↓)	1.27(↓)	1.09(↓)
C3	2	DE	2	2	0.36(↓)	1.09(↓)	2.07(↑)	1.21(-)
C4	6	AB	5	5	0.74(↑)	1.12(-)	1.71(↑)	1.20(↓)
C5	6	C2	5	3	0.82(↑)	1.35(↑)	1.41(-)	1.32(↑)
C6	3	C2	4	6	0.55(-)	1.28(↑)	1.57(↑)	1.29(↑)
C7	2	DE	1	1	0.34(↓)	1.32(↑)	1.05(↓)	1.27(↑)

Social Class: AB-F from high to low; Income Level: 1-5 from low to high, and 6 represents unknown; '-': median; '↑': higher than the median; '↓': lower than the median.

As with cluster 5, cluster 6 shares the same social class label but has fewer rooms and residents. This cluster has high IVSs in all three periods, indicating that participants are less price-sensitive. Cluster 3 has the highest IVS during peak times; however, it has a low average daily load consumption compared to other clusters. Households in cluster 3 have fewer adults and rooms, which may explain the low average electricity consumption throughout the day.

On the other hand, cluster 3 has a lower social class, and residents of this cluster tend to follow their daily routines, commute to work during the day, and return to their homes after working during peak hours. Consequently, Time-of-Use tariffs do not promote a peak shift in peak demand for individuals in this cluster since electricity is the rigid demand during peak hours.

In clusters 1 and 7, the average daily load is relatively low, particularly in cluster 1, with only 0.16 KWh per day. There is also a low IVS at peak times for these two clusters, indicating that the participants are price-sensitive to the Time-of-Use tariff. This may be due to the lower income levels of the participants in these clusters.

- **Multi-Scale Dynamic Profiling Results**

  There are two ways the new daily load curve can be generated: at the individual level (existing or new participants) or at the regional level. When a new daily load curve is observed, the multi-scale dynamic profiling modules will adaptively classify it into one of the universally-sketched clusters. The outputs of this module consist of two parts: the refined social information sketch and the multi-scale dynamic cluster trajectory.

  Table 4.5 provides the social information sketches in the loose sketch layer, followed by the detail refinement layer that includes the social sketches from the updated clusters of the existing participants and the new participants for each cluster. Based on the social information sketch in the loose sketch layer, the refined social information sketch is generated by a detailed refinement process, resulting in a richer social information profile for different electricity consumption patterns. This information can be used to determine characteristics of electricity consumption among participants with different socioeconomic backgrounds.

  As more participants join a cluster, more types of social information will be added. The final sketch can be defined by summarizing the social information from the updated clusters. According to table 4.5, most loose sketched clusters have been enriched except for cluster 5, which has a medium income level and social class, with 5 adults and 6 rooms. Furthermore, we expect that all final refined details will fall within a narrow range since the dynamic profiling module refines details based on universally-sketched clusters, and we do not wish the final refined cluster to include participants from a variety of social backgrounds.

  Clusters 1, 3, and 7 have similar final sketches since they all have fewer residents, rooms, and a lower social class and income level. Since more participants are clustered in cluster 1, more types of room numbers have been added. Also, cluster 3 has been enriched regarding the types of residents, room numbers, and income levels. The social information for cluster 7 has been refined except for the income level.

  The participants in clusters 2 and 4 have a high social class and income level, and only the adult resident numbers have been refined for both clusters. Furthermore, it is important to note that the income level of cluster 6 is unknown at the loose sketch layer, and this information is then revealed by the refined dynamic profiling module. Using refined dynamic profiling, it is possible to complement the loose sketches derived from layer 1 with additional informa-

Table 4.5: Refined social information sketch

	Level	No. Adults	No. Rooms	Social Class	Income Class	Final Defined Sketch
C1	Loose sketch	1	1	F	1	1 adult, 1-2 rooms, social class F, income level: 1
	Existing	1	2	F	1	
	New	1	1	F	1	
C2	Loose sketch	5	5	AB	5	5-6 adults, 5 rooms, social class AB, income level: 5
	Existing	6	5	AB	5	
	New	5	5	AB	5	
C3	Loose sketch	2	2	DE	2	1-3 adults, 1-2 rooms, social class DE, income level: 1-2
	Existing	3	2	DE	1	
	New	1	1	DE	1	
C4	Loose sketch	6	5	AB	5	5-6 adults, 5 rooms, social class AB, income level: 5
	Existing	5	5	AB	5	
	New	6	5	AB	5	
C5	Loose sketch	6	5	C2	3	6 adults, 5 rooms, social class C2, income level: 3
	Existing	6	5	C2	3	
	New	6	5	C2	3	
C6	Loose sketch	3	4	C2	6	3 adults, 4 rooms, social class C2, income level: 3
	Existing	3	4	C2	3	
	New	3	4	C2	3	
C7	Loose sketch	2	1	DE	1	1-2 adults, 1/3 room, social class DE/F, income level: 1
	Existing	1	1	F	1	
	New	1	3	F	1	

Social Class: AB-F from high to low; Income Level: 1-5 from low to high, and 6 represents unknown; '-': median; '↑': higher than the median; '↓': lower than the median.

tion based on the updated information.

By analysing the multi-scale dynamic cluster trajectories, the system tracks the changes in electricity consumption behaviour at different levels, which is essential for developing dynamic pricing strategies and enabling demand side management [128]. Moreover, the multi-scale cluster trajectories in the Universal-OFA can have varying time scales (e.g., daily, weekly, monthly, yearly), depending on the specific requirements of the system. In our experiment, we use the monthly time scale as a showcase by selecting a representative cluster based on the cluster with the maximum mode in each testing month for each participant. As a result, we can obtain a total of 3440 monthly dynamic trajectory records at the individual level (1717 existing and 1723 new participants) and a monthly dynamic trajectory record at the regional level.

Based on the monthly dynamic trajectories, 628 existing and 645 new participants showed stable cluster trajectories, i.e., their load usage patterns remained in the same cluster throughout the testing period. The demand side management strategy can be optimized for these participants based on the intra-day load usage pattern sketches of the corresponding cluster in Table 4.4. For the rest participants, there are 618 existing participants and 619 new participants at the individual level who are experiencing a cluster shift between May and June (the seasonal transition months), which requires additional attention as distinct load usage patterns may be generated when cooling appliances are used.

Figure 4.6 gives the region level dynamic cluster trajectory and an example of the individual level dynamic cluster trajectories.

According to the dynamic cluster trajectory, individual level participant 1059 remained in cluster 4 from April to June but shifted to cluster 2 between June and July. Based on the intra-day load usage pattern sketches for clusters 2 and 4 in Table 4.4, it can be concluded that from April to June, participant 1059 had high load usage during peak times. There has been an average daily electricity consumption increase after July, despite a decrease in fluctuations in peak electricity consumption. During July, high temperatures may have affected the load usage behaviour of the residents in this household, increasing load consumption since cooling appliances are used continuously.

At the regional level, however, the load consumption pattern has remained in cluster 4 for the past four months, indicating that load consumption during peak hours is volatile and that most participants in this region are not sensitive to tariff changes. It is, therefore, necessary to adjust the electric-

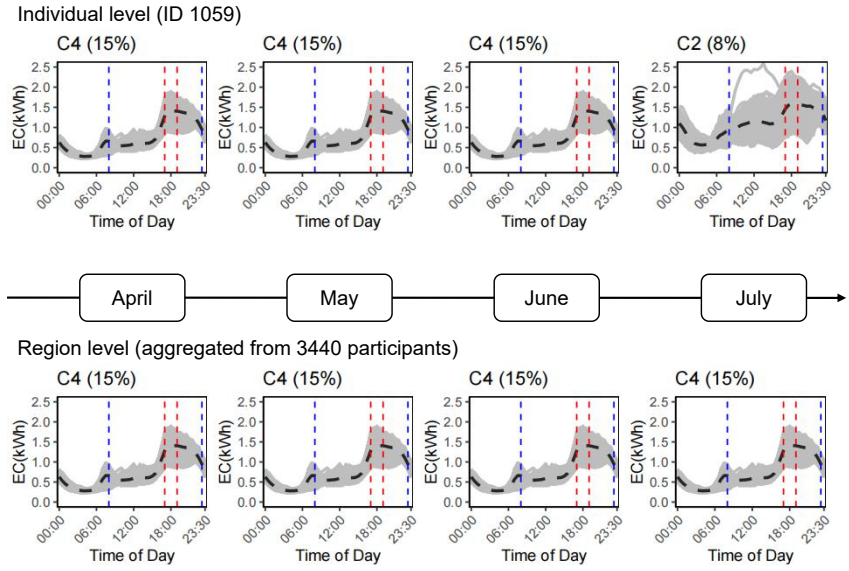


Figure 4.6: A dynamic cluster trajectory example at the individual levels (upper) and the dynamic cluster trajectory at the region level (lower).

ity pricing strategy to achieve a more successful result regarding demand-side management.

4.3.3.3 Multi-scale online load forecasting

- **Performance Evaluation Indices**

  To measure the performance of the proposed framework, for each forecasted daily load curve containing 48 half-hourly data points, we use the MAE defined on each day to measure the daily errors of the forecasted results:

$$MAE(d) = \frac{1}{P}\sum_{i=1}^{P}\left\{|\hat{L}_{n_d}(t_i) - L_{n_d}(t_i)|\right\}, \tag{4.11}$$

  where $\hat{L}_{n_d}(t_i)$ is the predicted value at the i-th half-hour in day d regarding to the real value $L_{n_d}(t_i)$. Another common evaluation criterion is the MAPE, which is calculated by dividing the MAE by the real load value. However, in our case, the real value $L_{n_d}(t_i)$ can be (near) zero since there are periods when the customer does not use any electricity. We, therefore, do not consider the accuracy metric of MAPE in this study.

In addition, the improvement rate (IR) is introduced as a means to measure the superiority of the proposed framework over other benchmark models [230]:

$$IR_{MAE} = \frac{MAE_B - MAE_T}{MAE_B} \times 100\%, \tag{4.12}$$

where IR_{MAE} denotes the improvement rate of the target model MAE_T over its benchmark model MAE_B in terms of MAE, the target model will have a higher level of prediction accuracy if the IR is positive. In the following experiments, the results from the proposed Online-FDA are set to be the target model MAE_T and compared with the MAE_B of other benchmark scenarios.

- **FDN Models Performance Benchmarking**

  Based on the previous studies [63, 230, 94, 220, 133, 22, 151], we now compare the proposed Universal-OFA model with various state-of-the-art approaches without real-time feedback adjusting, including the KNR model, SVR model, and the LSTM model. Considering that these models are trained on all the 3440 participants, the Universal-OFA model is also tested using the same offline data settings (trained in the loose sketch layer without utilising the detail refinement layer) by developing models based on the load usage patterns of all the participants rather than dividing them into the existing and new participants. Each model runs the testing procedures five times, and the final score is the average of all runs. Table 4.6 computes the mean MAE scores and the IR_{MAE} of 100 testing days (from April 07, 2010, to July 14, 2010) for these models, and Fig 4.7 presents the detailed distribution of the MAE scores for these models during the testing period.

Table 4.6: Evaluation mean scores and the improve rate of 100 days for different models

	KNR	SVR	LSTM	Universal-OFA
MAE	0.227	0.219	0.201	**0.188**
IR_{MAE}	17.18%	14.16%	6.47%	\

As shown in Table 4.6, the proposed Universal-OFA has the lowest MAE compared to other models evaluated in the experiment, outperforming the KNR model with an IR of 17.18%. Moreover, the proposed Universal-OFA improves forecasting accuracy by 6.47% over the LSTM model. Further, the SVR and LSTM perform better than the KNR models because they are more robust to noisy data and can identify load usage patterns for unseen load parameters.

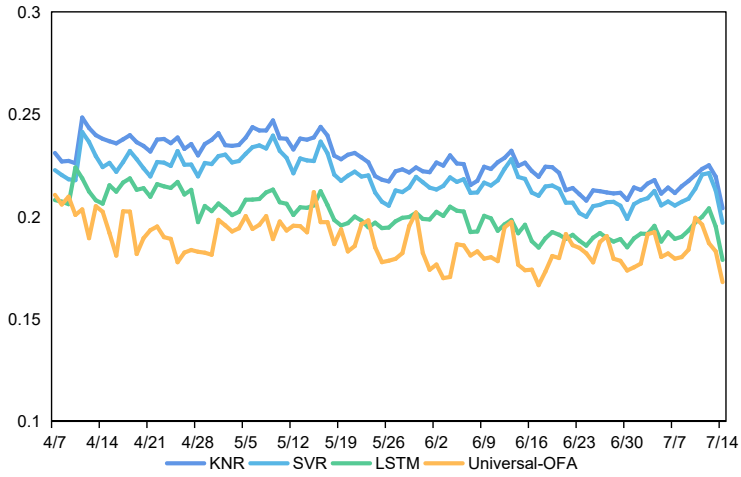


Figure 4.7: MAE scores for forecasted daily load of 100 days of different models.

Moreover, as shown in Figure 4.7, compared with other models, the MAE scores of the Universal-OFA for forecasting load in April are lower, demonstrating that the proposed FDN structure of Universal-OFA can assist in capturing the characteristics of the load curves, and thereby produce more accurate forecasts.

Despite May and June being unseen months for the forecasting models, the results show that Universal-OFA achieved the best performance during these months with the lowest MAE scores On the other hand, it is important to point out that the scores oscillate regularly, and most of the spikes occur on Saturday, which is a transition day between weekdays and weekends. This may be because the Universal-OFA needs help capturing the pattern shifts in electricity consumption behaviour between weekdays and weekends, thus leading to relatively low forecasting accuracy, even though it is still better than other models. Hence, it might be worthwhile to explore the possibility of providing a scheme that would enable the Universal-OFA to promptly track pattern shifts between weekdays and weekends.

- **Online FDN Models Forecasting Results**

  Following the functional clusters universal-sketching module, seven universally-trained FDN models were developed using historical load records of each cluster.

  After the dynamic profiling, the MLP regression head of the corresponding

universally-trained model will be dynamically updated to forecast the load at different levels for the testing period. The same updating step will be repeated 100 times to get the forecasts for 100 days.

In the multi-scale online FDN model updating phase, for one day, each online FDN model updating step for the newly arrived smart meter readings takes about 2.6s. The models ran the testing/updating procedures five times and used the mean values as the final scores. We evaluate the effectiveness of the proposed Universal-OFA model by comparing the following scenarios:

- Individual-OFA: Universal-OFA model trained on the 1717 individual level participants in the loose sketch layer and further refined on the real-time load sequences of the existing participants.
- Individual-OFA$_{offline}$: Universal-OFA model trained on the 1717 individual level participants in the loose sketch layer and tested on the real-time load sequences of the existing participants, i.e., without utilising the detail refinement layer.
- Individual-OFA$_{new}$: Universal-OFA model trained on the 1717 individual level participants in the loose sketch layer and further refined on the real-time load sequences of the newly joined participants.
- Regional-OFA: Universal-OFA model trained on the 1717 individual level participants in the loose sketch layer and further refined on the real-time aggregated load sequences of the 3440 participants at the regional level.

The Individual-OFA$_{offline}$ is developed without utilising the information from the newly arrived daily load records to evaluate the improvement of the detail refinement layer in the overall Universal-OFA framework. Moreover, the Individual-OFA$_{new}$ is developed to explore the generalisation ability of the universal models developed in the loose sketch layers at the individual level, and the Regional-OFA is developed to assess the generalisation ability of universal models developed in the loose sketch layers at the region level.

The final MAE scores of each model for each cluster in the above scenarios are the mean scores over 100 days, as shown in Table 4.7.

Overall, the forecasting results at both levels of the Universal-OFA are promising when compared to other scenarios, with the lowest mean MAE of 0.161 at the individual level and 0.070 at the region level. The noteworthy point is that the Individual-OFA$_{new}$ achieved the same mean MAE score as the Individual-OFA, indicating that universal models developed in the loose sketch layer of

Table 4.7: The MAE score of each cluster in different forecasting scenarios

	Individual-OFA	**Individual-OFA$_{offline}$**	**Individual-OFA$_{new}$**	**Regional-OFA**
C1	0.041	0.046	0.044	0.032
C2	0.179	0.183	0.183	0.073
C3	0.168	0.171	0.168	0.072
C4	0.189	0.193	0.190	0.016
C5	0.193	0.196	0.186	0.182
C6	0.186	0.188	0.186	0.059
C7	0.171	0.175	0.172	0.053
Overall	0.161	0.166	0.161	0.070

the Universal-OFA can be generalized at the individual level robustly. Moreover, the Regional-OFA achieved the best performance with a forecasting MAE of 0.070, indicating that aggregation reduces the inherent variability in load curves leading to a smoother profile of load consumption. Meanwhile, since FDN$_{offline}$ was trained without a refinement layer, it performs poorly compared to other scenarios, indicating that the Universal-FDA can continuously update itself in response to dynamic fluctuations in load data by incorporating real-time feedback from smart meter recordings.

Most cluster models within the Individual-OFA framework have shown the best performance for individual level forecasting. Individual-OFA$_{new}$ models for clusters 3 and 6 were found to achieve the same MAE as Individual-OFA models, and Individual-OFA$_{new}$ models for cluster 5 even outperformed Individual-OFA models. These models exhibit clear peaks in their universally sketched cluster results (Figure 4.5), demonstrating the ability of Universal-OFA to capture fluctuations in load usage patterns and its superior ability to generalize.

Furthermore, to better understand the half-hourly MAE distribution and demonstrate the advantage of the proposed framework, Figure 4.8 shows the MAE distribution for the last run of both the Individual-OFA forecasts and the Individual-OFA$_{offline}$ forecasts for 100 consecutive days. The daily forecasts show that the MAE scores of the proposed Individual-OFA framework decrease as the number of

days increases, whereas it does not happen in the Individual-OFA$_{offline}$ forecasts, implying that by incorporating real-time feedback from smart meter recordings, the Individual-OFA is able to learn from the newly arriving smart meter data to continuously update itself in response to dynamic fluctuations in load demand data.

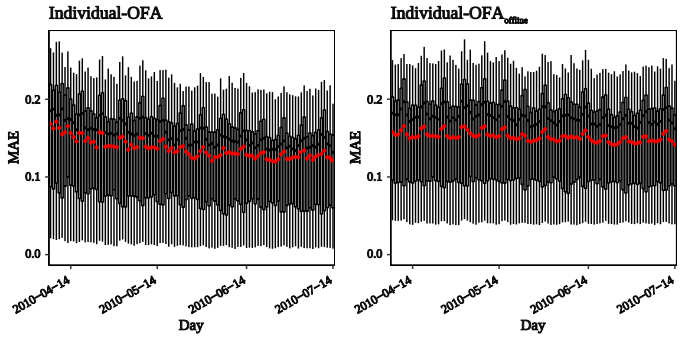


Figure 4.8: Box plot of the MAE scores for 48 half-hourly forecasted daily load of 100 unseen days of the Individual-OFA and Individual-OFA$_{offline}$. The black short lines are the daily median MAE and the red dots are the daily mean MAE.

4.3.4 Summary

Three significant conclusions can be drawn from the discussion of the experiment results above: 1) The Universal-OFA framework has been demonstrated to be universally applicable to the dynamic profiling and forecasting of multi-scale functional load, especially at higher aggregation levels. 2) IVSs enhance the robustness of the profiling process, enabling effective tracking of changes in load usage behaviour and analysis of the relationship between social information and the load profile. 3) The use of real-time daily load feedback renders the proposed Universal-OFA highly promising for forecasting short-term real-time load compared to various state-of-the-art models.

4.4 Chapter Summary

We propose a novel two-unit Universal-OFA model for the dynamic profiling and forecasting of multi-scale load, which is enhanced by a loose sketch-detail refinement strategy. The loose sketch layer enables the development of universal models, while the detail refinement layer allows the details to be refined based on data

obtained from multi-scale load. In the proposed model, an intra-day volatility score was proposed to enhance the multi-scale load dynamic profiling unit. Moreover, both Universal-OFA units train loose sketch layers by collecting historical load records anonymously into a common pool for undifferentiated clustering and training, ensuring customer identity privacy while providing flexibility for multi-scale load forecasts. By taking into account real-time feedback from newly arrived load records, the Universal-OFA enables dynamic tracking and forecasting of load usage behaviour at different levels. Experiments with real-world data have shown that the IVSs can boost the multi-scale load dynamic profiling unit in the proposed Universal-OFA, which can track the dynamic changes of the multi-scale load profiles and sketch the social information for the load profile at the individual level. Furthermore, quantitative comparisons have demonstrated the superiority of the proposed Universal-OFA in forecasting different levels of daily load.

Chapter 5

Distributed and Privacy-preserving Machine Learning Framework for Smart Meter Data Applications

5.1 Introduction

The NILM can help analyze the electricity consumption behaviours of users and enable practical smart energy and smart grid applications. As compared with data collected at the individual and other higher aggregation levels, smart meter data at the appliance level is more susceptible to leakage since they reveal the specific load usage patterns of residents. On the other hand, smart meters are privately owned and distributed, which makes real-world applications of NILM challenging. To this end, instead of analysing smart meter data at the individual level and the higher aggregation level as in Chapter 3 and 4, this chapter develops a distributed and privacy-preserving federated deep learning framework for NILM. Specifically, this framework is separated into two modules: the *FederatedNILM* and the *DP*^{2}*-NILM*, which are adapted from our paper 2. The *FederatedNILM* considers the distributed learning for smart meter data analysis, and the *DP*^{2}*-NILM* provides enhanced utility optimization and privacy-preserving techniques based on the *FederatedNILM*.

The rest of this chapter is organized as follows. Firstly, the problem statement is given in Section 5.2. Secondly, section 5.3 provides the preliminaries used in *FederatedNILM* and *DP*^{2}*-NILM*. Thirdly, Section 5.4 describes the *FederatedNILM* module, in which subsection 5.4.1 presents the designed *FederatedNILM* system and

details its implementation process, and subsection 5.4.2 conducts the performance evaluation based on the comparative experiments. Similarly, Section 5.5 describes the *DP*^{2}*-NILM* module. Specifically, subsection 5.5.1 overviews the three-tier workflow of *DP*^{2}*-NILM*, and subsection 5.5.2 and 5.5.3 details the utility optimizations schemes and the privacy-preserving schemes of *DP*^{2}*-NILM*. Then, the performance evaluations on real-world datasets are conducted in subsection 5.5.4. The final chapter conclusion is given in Section 5.6.

5.2 Problem Statement

Modern urbanization, lifestyles, and technological advancements have increased the energy demand. Energy supply generates greenhouse gas emissions that accelerate climate change, which presents a significant threat to the security and prosperity of the global community [73]. In the UK, legal obligations regarding climate change have been enacted, putting increased strain on the traditional centralized power grid [36]. The smart grid has been brought up using information systems to create a more reliable and intelligent power grid network [114], which has the potential to contribute to the decarbonization of the energy system and is a leading candidate for renewable energy sources [36]. As a key part of a smart grid, smart meters allow NILM to help smart meter clients reduce energy consumption by scheduling appliance usage hours and monitoring abnormal electricity usage patterns.

Recent studies have proposed many novel NILM frameworks based on deep learning. Although deep learning models usually perform well on NILM, such models still face several challenges. The number of the labelled data generated by a single household is limited, and the size of the training set has a great impact on the effectiveness of the deep learning model. Therefore, it is necessary to collect the labelled data from multiple data sources on the premise of ensuring data security. Besides, different users have different lifestyles and thus have different electricity usage patterns (i.e. data distribution), which put forward higher requirements for the generalization ability of the NILM model. Moreover, given the increasing public attention to data privacy and security preservation, it is necessary to satisfy the needs of training models with not only high precision but also reasonable communication efficiency under the premise of ensuring individual data privacy.

To overcome the above challenges, federated learning [143] was proposed where private data of individual users do not need to be uploaded to a central server for centralized training. Instead, under the coordination of the central cloud server, each participant can carry out the model training locally and only exchange typical parameters of their local model such as updated gradients [80]. Compared with

other privacy-preserving technologies that need to encrypt the original data set, federated learning does not need to collect the original data centrally, therefore model training in this framework does not involve data transmission and public sharing, and can thus help to achieve individual data privacy protection. On the other hand, the wide penetration of the IoT in various areas [198] has sparked new opportunities for federated learning by providing massive amounts of distributed user-generated data on intelligent IoT devices and applications, which is poised to make substantial contributions in all aspects of our modern life, such as smart healthcare and smart grid system [15].

Although federated learning has been studied in various areas, limited attention has been paid to smart grid applications especially for NILM. Since NILM is one of the key technologies to unlock the full potential of local and distributed energy resources, a distributed and privacy-preserving framework is urgently needed to enable its practical applications. To this end, the first part of this chapter develops a distributed and privacy-preserving federated deep learning framework for NILM based on federated deep learning.

Moreover, it is worth noting that there lacks comprehensive research exploring the utility optimization schemes and the privacy-preserving schemes in different FL-based NILM application scenarios. Therefore, in the second part of this chapter, we make the first attempt to conduct FL-based NILM focusing on both the utility optimization and the privacy-preserving by developing a distributed and privacy-preserving NILM framework and carrying out comparative experiments on practical NILM scenarios based on real-world smart meter datasets. Specifically, two alternative federated learning strategies are examined in the utility optimization schemes, i.e., the FedAvg and the FedProx. Moreover, different levels of privacy guarantees, i.e., the local differential privacy federated learning and the global differential privacy federated learning are provided in the DP^2-NILM.

5.3 Preliminaries

We introduce several essential concepts related to the proposed framework in this section.

5.3.1 Federated deep learning

When data owners intend to combine their local data with training a common utility model, the traditional centralized approach pools their private data at a central server, during which data uploading and integration process are often restricted by data privacy legislation. To address this challenge, FL was brought up [116],

which only requires the exchange of updated model parameters rather than the raw data between clients and the central server, and therefore is deemed to be the state-of-the-art approach for distributed data privacy protection.

FL is a machine learning strategy aimed at training a high-quality global model while the raw private datasets are distributed locally in each client without transferring them to a central server. The training process of the federated deep learning framework is shown in Figure 5.1, which can be described in three steps.

- **Step 1.** Each client trains their local model and updates model parameters during each training round. Then, each client passes the updated parameters to a central server.
- **Step 2.** The global model aggregates the updated parameters from all local clients and updates its parameters accordingly in the central server.
- **Step 3.** The updated global model parameters are then broadcast to each local client, and these three steps are iterated for multiple rounds until convergence is reached.

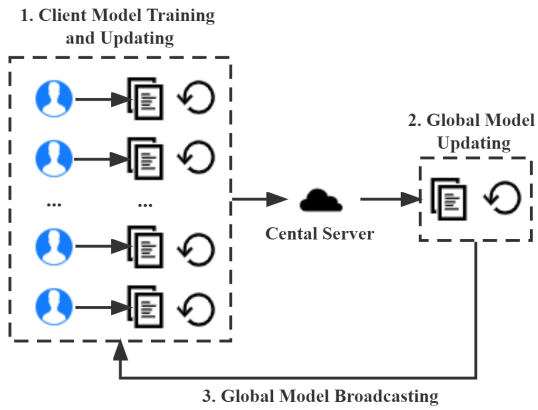


Figure 5.1: Training process of the federated deep learning framework.

5.3.2 Differential privacy

DP introduces noise into the raw dataset to provide statistical guarantees against the information a malicious adversary may infer from the output of a randomized algorithm [56].

Definition 1 (**Differential Privacy** [4])**.** A random algorithm $\mathcal{M}$ is compliant with (ϵ, δ)-DP if for any two neighbouring input datasets L, L' and for any subset of outputs/events $\boldsymbol{S} \subseteq Rang(\mathcal{M})$,

$$\Pr[\mathcal{M}(L) \in \boldsymbol{S}] \leq e^{\epsilon} \Pr[\mathcal{M}(L') \in \boldsymbol{S}] + \delta. \tag{5.1}$$

In the above equation, ϵ is the privacy budget/loss, which is inversely proportional to the privacy level. δ is the probability that the upper privacy bound is broken, i.e., the occurrence of a bad event. It is a plain ϵ-DP when δ equals 0.

For a real-valued function $\mathcal{F}$, a common exemplification is to calibrate an additive zero-mean Laplacian or Gaussian noise mechanism to the sensitivity of $\mathcal{F}$, which can be denoted as

$$\Delta\mathcal{F} = \max_{L,L'} \left\| \mathcal{F}(L) - \mathcal{F}(L') \right\|_1 . \tag{5.2}$$

Depending on whether a single record is included or excluded, the sensitivity $\Delta\mathcal{F}$ measures the maximum change in output.

The Gaussian mechanism adds Gaussian noises to $\mathcal{F}$ to satisfy (ϵ, δ)-DP: $\forall\delta \in (0, 1)$, the noise is denoted by $\mathcal{N}(0, \Delta\mathcal{F}^2 \cdot \sigma^2)$, and we have

$$\mathcal{M}(L) = \mathcal{F}(L) + \mathcal{N}(0, \Delta\mathcal{F}^2 \cdot \sigma^2), \tag{5.3}$$

where $\Delta\mathcal{F} \cdot \sigma$ is the standard deviation, and $\sigma \geq \frac{\sqrt{2\ln(1.25/\delta)}}{\epsilon}$.

5.4 FederatedNILM: A Distributed Smart Meter Analysis Framework

5.4.1 Framework Design and Implementation

5.4.1.1 The workflow of the proposed FederatedNILM framework

The FederatedNILM framework aims to accurately infer the ON/OFF states of multiple appliances in individual households in a distributed and privacy-preserving way. The whole workflow of the FederatedNILM can be described in three stages (see Algorithm 2).

- **Initialisation.** The parameters of global deep learning model w_G, global sharing batch B_G, along with parameters of model training including learning rate η, momentum ρ and loss function $\mathcal{L}$ need to be initialised first. Then set the global and local communication round index to 1, initialise the moment estimate variable v to 0, and begin the first round of training.

Algorithm 2: FederatedNILM for multiple households

Input: Aggregated load consumption of target appliances from all N houses $\{L_n\}, n = 1, 2, \ldots, N$, the number of global communication rounds R_G, the global sharing batch B_G, the local batch B_L, the number of local epochs R_L

Output: The optimal global deep learning model parameters w_G^*.

Initialization:
The initial global deep learning model parameters w_G, learning rate η, momentum ρ, loss function $\mathcal{L}$, the global sharing batch B_G;
The global communication round index $r_G = 1$;
The local epoch index $r_L = 1$;
The moment estimate variable $v = 0$;
Procedure:
// Global deep learning model aggregation, training, and broadcasting
for $r_G \leq R_G$ **do**
    for $n = 1, 2, \ldots, N$ ***in parallel*** **do**
        $w^n_{r_G+1} \leftarrow HouseholdsUpdate(w^n_{r_G})$;
    end
    $w_{r_G+1} \leftarrow \frac{\sum_{n=1}^{N} w^n_{r_G+1}}{N}$;
    Replace the old global deep learning model with the new parameters, which are stored in the global sharing batch B_G: $w_{r_G} \leftarrow w_{r_G+1}$;
    $r_G \leftarrow r_G + 1$.
end
return The global deep learning model with parameters w_{R_G}
// Local households model updating, training, and uploading
$HouseholdsUpdate(w^n_{r_G})$:
Split L_n into batches of size B_L;
while $r_L \leq R_L$ **do**
    for *each batch of* L_n **do**
        Calculate the gradient by $d \leftarrow \nabla w^n_{r_G} \mathcal{L}$;
        Update biased moment estimate variable by $v \leftarrow \rho v + d$;
        Update the local model parameters by $w^n_{r_G} \leftarrow w^n_{r_G} - \eta v$;
    end
    $r_L \leftarrow r_L + 1$.
end
return $w^n_{r_G}$

- **Local households model updating, training, and uploading.** The local households will train their own local deep learning model based on the local data L_n after receiving the broadcast parameters from the global deep learning model. As described in $HouseholdsUpdate()$, in each local epoch, the local deep learning models aim to find the best approximation F_s (Equation (3.4)). The local training returns the updated parameters from all the local models, which are then uploaded to the global cloud server.

- **Global deep learning model aggregation, training, and broadcasting.** The global deep learning model receives the locally updated parameters from the global sharing batch B_G and adopts federated averaging (FedAvg) to the

parameter sets [143]. Then, the global deep learning model will be updated based on the FedAvg results. After this, the updated global parameters will be broadcast to the local models of each house. After running all the global communication rounds between the local households and the central cloud server, a final global deep learning model will be generated.

5.4.1.2 The deep learning model for NILM

In Chapter 3, we have used the sequence-to-point architecture to build NILM model, which is a suitable choice for applying transfer learning and building personalized NILM models for different appliances. However, in this chapter, the main focus of using the deep learning model is to enhance the overall performance of the NILM models, which in turn expands the choice of models. The deep learning architecture utilized to enhance the overall inferring performance in this chapter is inspired by Zhao et al. [237], which was originally used for image semantic segmentation. The selection of the above particular architecture is motivated by its potentially promising performance on NILM after appropriate adjustments as demonstrated in [167]. The complete layout of our chosen deep learning model for NILM is shown in Figure. 5.2

Specifically, the architecture of the deep learning model for NILM is composed of three modules: the encoder, the temporal pooling module, and the decoder.

- **Encoder.** The input of the encoder is the household aggregated load consumption of the target appliances over a 126 minutes interval. The encoder increases the space of features from a single aggregation value to 256 while paying the price of decreasing the time signal resolution by ten times.

- **Temporal pooling.** The temporal pooling module consists of four average pooling modules. The filters in this module are reduced from the whole size of the input signal to one-sixth of it, which is the same case with the stride. After going through a convolutional layer, the feature dimension of the input is reduced to a quarter of its original size, and the acquired feature maps were upsampled to increase their size to the size of the input time signals. Then the upsampled feature maps (shallow features) are concatenated with the original input signal (deep features) from the temporal pooling to get the final feature maps. The fusion of the deep and shallow features of the temporal pool could enable this block to get contextual information fed into the decoder.

- **Decoder.** The decoder receives the output from the temporal pooling block and passes it to a convolutional layer to recover the temporal resolution. Then

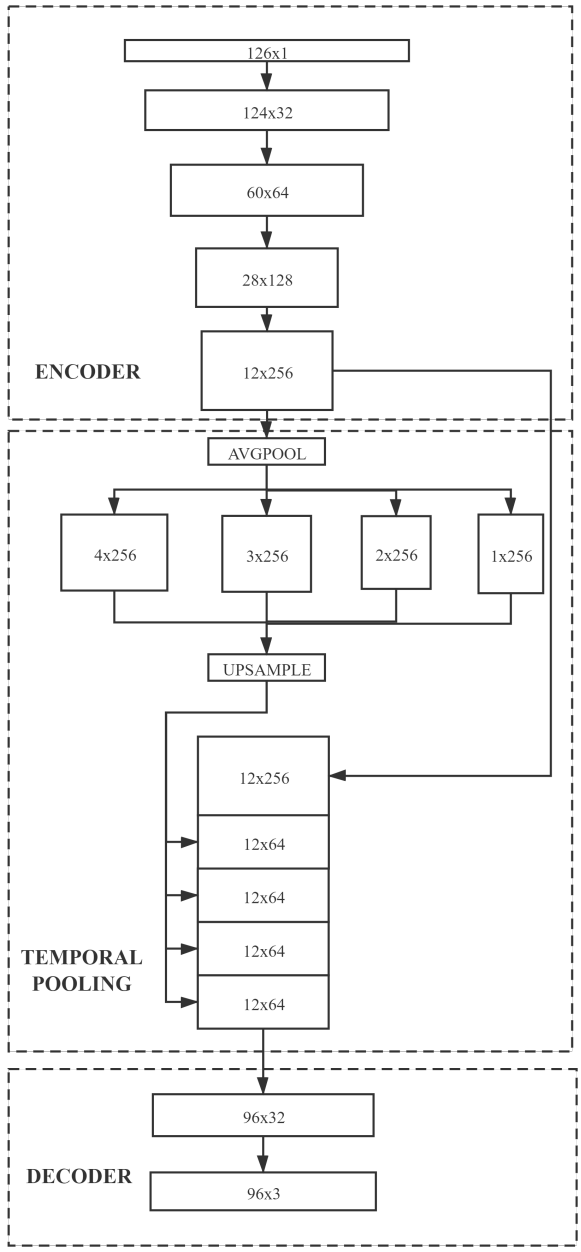


Figure 5.2: The overall layout of the deep learning model for NILM

the output is fed into the final convolutional layer to produce the final output. $Softmax$ is utilized to classify the states of multiple targeted appliances. After this, the binary cross-entropy is chosen as the loss function, which is given by:

$$\begin{aligned}\mathcal{L}^i_{class} =& \frac{1}{N}\sum_{n=1}^{N}\frac{1}{S}\sum_{s=1}^{S}(s^i_{n,s}\cdot\log p(s^i_{n,s}) \\ &+ (1-s^i_{n,s})\cdot\log(1-p(s^i_{n,s})))\end{aligned} \tag{5.4}$$

where N is the number of the training samples, S is the length of the output from $Softmax$ classifier, $s^i_{n,s}$ is the true state of i-th appliance, and $p(s^i_{n,s})$ denotes the predicted probability that i-th appliance is in the activation state. Also, stochastic gradient descent (SGD) is selected as the optimizer to facilitate the convergence of the binary cross entropy.

5.4.2 Evaluation and Discussion

This section gives numerical experiments to evaluate the proposed FederatedNILM framework. Firstly, experimental settings are introduced, including the environment setup, data preprocessing, training and testing split, and evaluation metrics. Secondly, we compare our proposed FederatedNILM with the centralized counterpart regarding model performance and training time efficiency. Finally, we compare the adopted deep learning architecture (used in the FederatedNILM and the centralized counterpart) with several existing advanced NILM models to evaluate the performance of the proposed FederatedNILM framework.

5.4.2.1 Experimental settings

The FederatedNILM model is implemented on Pytorch and conducted on a Windows 10 platform using an NVIDIA GeForce RTX 2080 Ti GPU with 64GB of RAM and CUDA v10.2. When conducting local model training and uploading, we assume that all clients are available. Since the entire project was completed on a single PC, the client models were updated consecutively within a set number of local epochs before communicating with the central server to train the final global model.

In the FederatedNILM, we consider the UK-DALE dataset [103], which has been introduced in subsection 3.3.4.2, Chapter 3. To demonstrate the effectiveness of the proposed FederatedNILM model and the utilized deep neural network architecture, we deploy a simplified experimental setup following Kelly and Knottenbelt [102] for comparison purposes. We consider common appliances possessed by most houses, i.e., fridge, dishwasher, and washing machine, which narrowed the dataset to houses

1, 2, and 5. The date range selected from the three houses is from 12/04/2013 to 01/07/2015 for house 1, 22/05/2013 to 03/10/2013 for house 2, and 29/06/2014 to 01/09/2014 for house 5.

Following the same data preprocessing procedure in [102], abnormal load consumption records were firstly filtered out by the max power threshold provided in Table 5.1 followed by a down-sampling of the aggregated load from 1s to 6s to align with the submetered data. The resampled data were normalized by subtracting the mean and then dividing a constant value 2000 W. Finally, the state series of each target appliance were derived from the activation-time thresholding as the input of the proposed FederatedNILM model where relevant thresholds are provided in Table 5.1.

Table 5.1: Relevant thresholds information

	Fridge	Dishwasher	Washing Machine
Max power (W)	300	2500	2500
Power threshold (W)	50	20	20
Min. ON duration (s)	1	60	60
Min. OFF duration (s)	0	60	5

In the experiment, both the seen house case and the unseen house case are considered to verify the effectiveness of FederatedNILM.

For the seen house case, the split of the three houses datasets is listed in Table 5.2. The first 80% series from each household was selected as the training set, followed by a 10% for validation and 10% for testing. In this case, the disaggregation ability of the model is evaluated when signatures of specific appliances are learned.

For the unseen house case, the choice of three houses allows us to train using two houses and test on another. The unseen house case aims to verify the generalisation ability of the model, and the generic signature characteristics of the same type of appliances need to be distinguished. As detailed in Table 5.3, we split two houses data into training and validation sets and used the other unseen house as the test set. Then, we average the results of all three different combination cases of the training and testing for comparative analysis.

The parameters used in the FederatedNILM model training are listed in Table

Table 5.2: Training, validating and testing splits for the seen house case model

	Train	Validation	Test
House 1	80%	10%	10%
House 2	80%	10%	10%
House 5	80%	10%	10%

Table 5.3: Training and testing splits for the unseen house case model

House	Case 1			Case 2			Case 3		
Number	Training	Validation	Testing	Training	Validation	Testing	Training	Validation	Testing
1	90%	10%	-	90%	10%	-	-	-	100%
2	90%	10%	-	-	-	100%	90%	10%	-
5	-	-	100%	90%	10%	-	90%	10%	-

5.4. Different global rounds [2, 4, 6, 8, 10] are selected to conduct comparative experiments. We repeat the experiment five times for all the cases and report the average performance for each model.

Four evaluation metrics, precision, recall, accuracy, and F_1, are used in the experiment to assess the performance of the proposed framework. The precision (eq. (3.11)), recall (eq. (3.12)), and F_1 (eq. (3.13)) has been introduced in subsection 3.3.4, Chapter 3. Recall the true positive as TP, true negative as TN, false positive as FP, and false negative as FN, the accuracy can be defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.5}$$

The accuracy reflects the ratio of all correctly identified samples to all the data sequences.

Table 5.4: Parameters used in FederatedNILM

Item	Explanation	Value
B_G	Global sharing batch size	32
R_G	Global communication rounds	[2, 4, 6, 8, 10]
B_L	Local batch size	32
R_L	Local epochs	10
η	Learning rate	1e-4
Activation function	-	ReLU
Dropout probability	-	0.1
ρ	Momentum	0.5
Optimizer	-	SGD

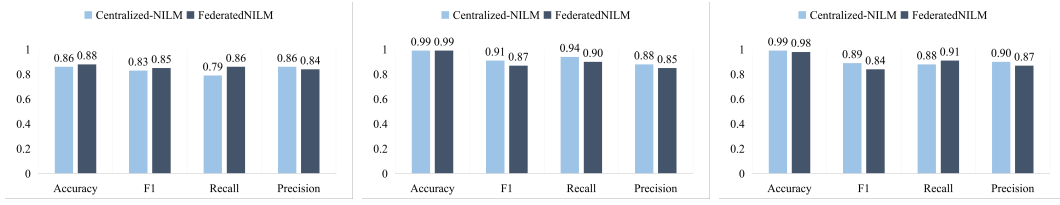
5.4.2.2 Comparative studies

We consider the FederatedNILM model and the centralized counterpart (termed as Centralized-NILM) for both the seen house case and unseen house case. For Centralized-NILM, each household needs to upload its original data directly to the trusted central server, in which the data are trained in a centralized way by the deep learning model described in Section 5.4.1.2. This scenario could not provide any data privacy guarantee to participants. FederatedNILM, on the other hand, does not require access to the original data. In this scenario, households only need to share the training outcomes (parameter sets) of the local deep learning model with the central server.

• Performance comparison with centralized model based on seen house case

In this section, we conduct comparative studies between the proposed FederatedNILM and Centralized-NILM for the seen house case. We run ten global rounds with ten local epochs for FederatedNILM training and 100 epochs for Centralized-NILM training.

Fig. 5.3 shows the disaggregation performance on the test dataset in the seen house case.



(a) Fridge (b) Dishwasher (c) Washing machine

Figure 5.3: Disaggregation performance on the test dataset in the seen house case.

From the results, we can see that for each appliance, both Centralized-NILM and FederatedNILM achieve satisfactory results on the dishwasher and washing machine and reasonable results on the fridge. It is worth pointing out that the fridge consumes relatively low power compared with other appliances and is likely to be learned with less evident signature during model training since its consumption can easily be omitted as unidentified load noise. This might explain why testing results of the fridge show relatively low scores compared with other appliances.

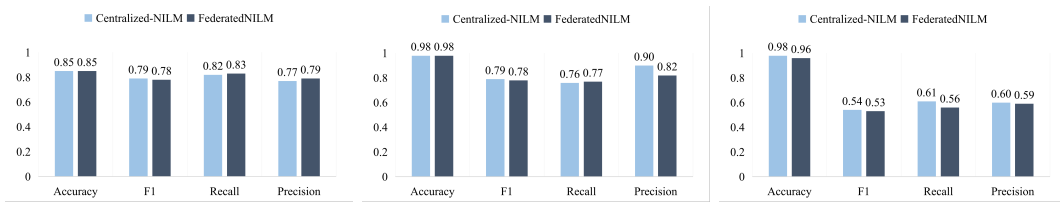
It should be highlighted that the FederatedNILM achieves very similar performance to Centralized-NILM on all appliances. Therefore, it is reasonable to infer that our proposed FederatedNILM framework works well in a distributed and

privacy-preserving manner for the seen house case and can achieve a good trade-off between data privacy and data utility.

• **Performance comparison with centralized model based on unseen house case**

In this section, we apply our proposed models for the unseen house case to evaluate the generalisation ability of FederatedNILM.

Figure 5.4 shows the disaggregation scores on the house not seen during training for the three target appliances.



(a) Fridge (b) Dishwasher (c) Washing machine

Figure 5.4: Disaggregation performance in the unseen house case.

Compared with the seen house case, both models produce satisfying results for the fridge and dishwasher but perform less well for the washing machine. One possible reason is that the washing machine, as a multi-state appliance, has more complex signature characteristics [57], which is likely to be more challenging to distinguish the generic signature characteristics.

We can also find that the FederatedNILM has very similar performance compared with Centralized-NILM on all appliances for the unseen house case, which aligns well with our conclusion on the proposed FederatedNILM framework in terms of data privacy and data utility in Section 5.4.2.2 for the seen house case.

• **Global round, local epochs, and training time efficiency**

In this section, different global rounds and their related training costs and testing results are given to explore the relationship between training efficiency and model performance.

The epochs for the Centralized-NILM are set to 20, 40, 60, 80, and 100, respectively, corresponding to the global rounds 2, 4, 6, 8, and 10 with ten local epochs of the FederatedNILM model. Tables 5.5 to 5.7 listed the testing results of both models using different global rounds in the unseen house case for fridge, dishwasher, and washing machine, respectively, and the best results of each model are marked in bold. Figure. 5.5 shows the training time of the two models with increasing epochs/global rounds.

Table 5.5: Test results for fridge

Global rounds	FederatedNILM Accuracy	Precision	Recall	F_1	Baseline epochs	Centralized-NILM Accuracy	Precision	Recall	F_1
2	0.80	0.80	0.77	0.70	20	0.80	0.76	0.73	0.75
4	0.82	0.69	0.76	0.74	40	0.81	0.78	0.73	0.75
6	0.83	0.72	**0.86**	0.78	60	0.83	0.79	0.74	0.75
8	0.84	0.70	0.82	0.73	80	0.83	0.80	0.73	0.79
10	**0.85**	**0.83**	0.79	**0.78**	100	**0.85**	**0.82**	**0.77**	**0.79**

Table 5.6: Test results for dishwasher

Global rounds	FederatedNILM Accuracy	Precision	Recall	F_1	Baseline epochs	Centralized-NILM Accuracy	Precision	Recall	F_1
2	0.79	0.55	0.54	0.45	20	0.97	0.70	0.74	0.74
4	0.81	0.58	0.57	0.68	40	0.97	0.74	0.81	0.73
6	0.98	0.73	0.69	0.69	60	0.98	0.74	0.84	0.77
8	0.89	0.75	0.77	0.69	80	0.98	0.76	**0.91**	0.77
10	**0.98**	**0.77**	**0.82**	**0.78**	100	**0.98**	**0.76**	0.90	**0.79**

The overall performance of the FederatedNILM model and the Centralized-NILM generally improves with the increase of the epochs.

With the increase of the global rounds, the time consumption cost also increases. The training time consumption of the FederatedNILM model is slightly higher but

Table 5.7: Test results for washing machine

Global rounds	FederatedNILM Accuracy	Precision	Recall	F_1	Baseline epochs	Centralized-NILM Accuracy	Precision	Recall	F_1
2	0.96	0.52	0.53	0.41	20	0.97	0.53	0.49	0.44
4	0.95	0.53	0.57	0.45	40	0.98	0.50	0.52	0.47
6	0.95	0.54	0.58	0.50	60	0.98	0.55	0.57	0.49
8	0.94	0.55	0.58	**0.54**	80	0.98	0.59	0.55	0.53
10	**0.96**	**0.56**	**0.59**	0.53	100	**0.98**	**0.61**	**0.60**	**0.54**

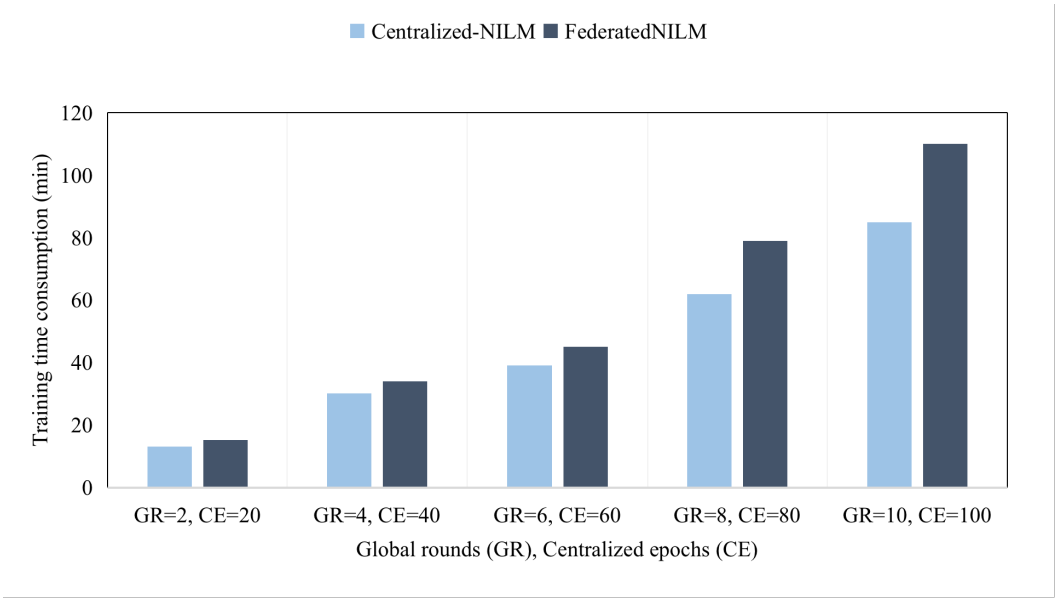


Figure 5.5: Training time for the proposed FederatedNILM and the centralized NILM model

still within the acceptable range compared with the Centralized-NILM. Besides, given similar classification scores, the proposed FederatedNILM model has similar training time efficiency compared with the Centralized-NILM. For instance, the proposed model obtained an accuracy, precision, recall, F_1 of 0.98, 0.77, 0.82, 0.78, respectively, for the dishwasher when global rounds reach ten. In contrast, similar scores (0.98, 0.76, 0.90, 0.79 for accuracy, precision, recall, F_1, respectively) were achieved in the Centralized-NILM with 100 epochs. Therefore, we can also infer that the training cost of FederatedNILM is within an acceptable range compared with the Centralized-NILM under similar accuracy requirements. The above analysis confirms that the proposed model could achieve a satisfactory trade-off between the computational costs and the model performance.

• **Performance comparison with state-of-the-arts**

In the previous subsections, we compare the proposed FederatedNILM with Centralized-NILM to examine its feasibility from the perspective of communication cost and model accuracy (i.e., distributed vs. centralized). In the following, we compare our adopted deep learning architecture (used in both FederatedNILM and Centralized-NILM) with state-of-the-art considered in [102] on model generalisation performance for the NILM task.

By considering the experiment environment set up in the proposed framework and in [102] and to allow for a direct comparison, the appliances and corresponding house numbers listed in Table 5.8 are considered.

Table 5.8: Training and testing houses for the unseen house case model

	Train	Testing
Dishwasher	[1, 2]	5
Washing machine	[1, 5]	2

Table 5.9 gives the comparative results of FederatedNILM and Centralized-NILM with state-of-the-art for dishwashers and washing machines, respectively. The models considered for the comparative study are combinatorial optimization (CO), factorial hidden Markov models (Neural-NILM FHMM), and deep neural network (Autoencoder, Rectangles, Neural-NILM LSTM, centralized-NILM, and FederatedNILM).

In general, the deep neural network-based models for NILM have relatively bet-

Table 5.9: Comparison results of Federated-NILM with state-of-the-arts

State-of-the-art models	Dishwasher				Washing machine			
	Accuracy	Precision	Recall	F_1	Accuracy	Precision	Recall	F_1
CO [102]	0.64	0.06	0.67	0.11	0.88	0.06	0.48	0.10
Neural-NILM FHMM [102]	0.33	0.03	0.49	0.05	0.79	0.04	0.64	0.08
Autoencoder [102]	0.92	0.29	**0.99**	0.44	0.82	0.07	**1.00**	0.13
Rectangles [102]	**0.99**	**0.89**	0.64	0.74	0.98	0.29	0.24	0.27
Neural-NILM LSTM[102]	0.30	0.04	0.87	0.08	0.23	0.01	0.73	0.03
Centralized -NILM	**0.99**	0.79	**0.99**	**0.83**	**1.00**	**0.85**	0.96	0.90
FederatedNILM	0.98	0.84	0.69	0.78	**1.00**	0.77	0.67	**0.92**

ter performance than other models. It is also worth pointing out that both the Centralized-NILM and FederatedNILM achieved better testing results on the unseen house 2 (see Tables 5.8 and 5.9) than the average testing results of all three cases in Table 5.3 for the washing machine. One possible reason is that house 1 provides the largest dataset, and when this house is considered an unseen house, as in Table 5.3, the training size of the model becomes much smaller, which could impact the model performance. Among all the methods, the Centralized-NILM model achieved the highest accuracy score for both appliances, which proves that the deep learning architecture we utilized in the proposed framework could enhance the local training performance and thus improve the overall state inference accuracy in our framework. As discussed in previous subsections, the above results also demonstrated that our proposed FederatedNILM with the same deep learning architecture as Centralized-NILM could achieve promising generalisation performance. Considering FederatedNILM works in a distributed and privacy-preserving manner, which is fundamentally different from the other considered methods (i.e., centralized-based methods), our proposed framework could achieve a good trade-off between data privacy protection and data utility.

5.5 DP^2-NILM: Providing Privacy Guarantee to Distributed Smart Meter Analysis

5.5.1 Framework Design Overview

5.5.1.1 Overview of DP^2-NILM

The key objective of our DP^2-NILM framework is to train different federated learning models focusing on various enhancement schemes, i.e., the utility optimization

schemes and the privacy-preserving schemes, for real-world NILM application scenarios. We further stress that the DP^2-NILM framework can easily incorporate various state-of-the-art DNN models and datasets. As presented in Figure 5.6, the whole workflow of the DP^2-NILM framework contains three tiers.

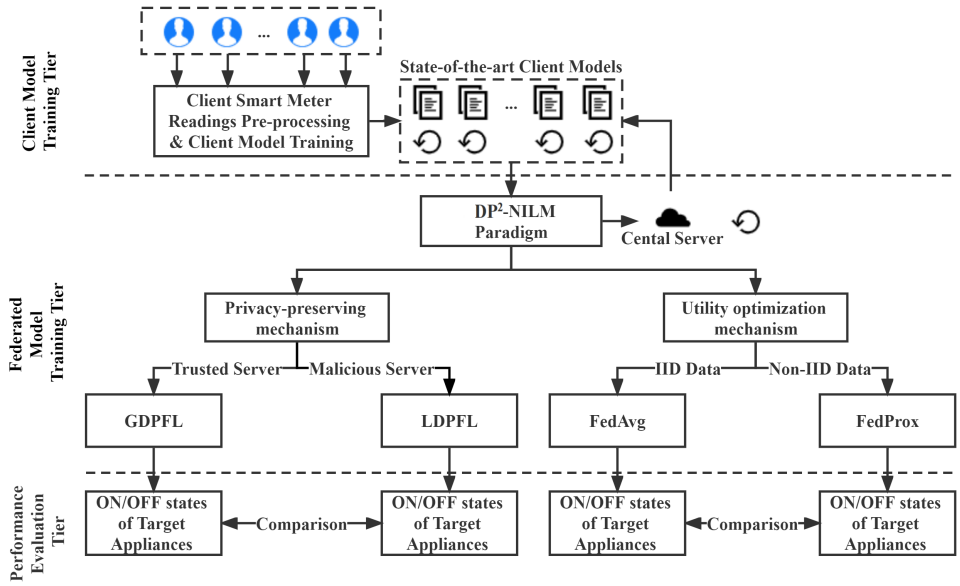


Figure 5.6: The workflow of proposed DP^2-NILM framework.

- **Client Model Training Tier.** In this tier, smart meter readings from the client side are preprocessed into standard formats for the federated pipeline. The client can either specify their privacy-preserving or data heterogeneity optimization requirements. After preprocessing, each client trains their data based on a state-of-the-art DNN model, which will be introduced in Section 5.5.1.2, and then upload their parameters through the DP^2-NILM paradigm.

- **Federated Model Training Tier.** This tier is the key part of the DP^2-NILM framework. Based on the particular requirement from the client model training tier, the DP^2-NILM assigns different federated learning mechanisms to each client. For example, a client-side requires a strict privacy-preserving mechanism to protect sensitive data. After receiving this request, DP^2-NILM will deliver a high-level privacy-preserving paradigm, the local differential privacy federated learning (Section 5.5.3.2), to train the FL model based on the typical FL training steps.

During FL training, the objective for the global model can be formulated as:

$$\min_{w_g} \mathcal{L}_g(w_g) = \frac{1}{|L|} \sum_{n=1}^{N} |L^n| \cdot \mathcal{L}_c^n(w_c), \tag{5.6}$$

where $L_g(w_g)$ is the loss of the global model, $\mathcal{L}_c^n(w_c)$ is the loss of the n-th local client model. Then, the initialized/updated server parameter w_g will be broadcasted to each client. The objective for each client can be formulated as:

$$\mathcal{L}_c^n(w_c) = \frac{1}{|L^n|} \sum_{i \in L^n} \mathcal{L}_i(w_i), \forall L^n \in L, n \in \{1, 2, \ldots, N\} \tag{5.7}$$

where $\mathcal{L}_i(w_i)$ is the loss of a single smart meter reading. Each household $n \in \{1, 2, \ldots, N\}$, and generates its private smart meter readings $L^n = \{(l^n, s^n), (l^n, s^n), \ldots, (l^n, s^n)\}$, where l^n is the aggregated load consumption of the target appliances, and s^n is the corresponding states (ON/OFF) set of these appliances.

The most commonly used optimization algorithm for FL is the FedAvg [143]. Based on the FedAvg, two subsequent research streams for enhancing the FL paradigm have been proposed, i.e., the utility optimization schemes and the privacy-preserving schemes. Following this development, the DP^2-NILM framework uses the FedAvg as the baseline to include the above two enhancing schemes. Specifically, the DP^2-NILM adopts the FedAvg and the FedProx to optimize the model utility for FL-based NILM. Furthermore, studies [208, 208] have provided clear theoretical foundations for GDPFL and LDPFL based on the FedAvg, and hence the DP^2-NILM develops GDPFL and LDPFL in privacy-preserving schemes based on FedAvg.

- **Performance Evaluation Tier.** We designed different model training paradigms for different NILM application scenarios based on three real-world smart meter datasets, and the model performance of each scenario is evaluated and validated in this tier.

5.5.1.2 State-of-the-art NILM client model

We introduce a state-of-the-art deep learning architecture, i.e., the pyramid scene parsing network (PSPNet) [237], to enhance the performance of DP^2-NILM in the local client model training and the central server global model training, which was originally used for image semantic segmentation. The selection of this particular architecture is motivated by its potentially promising performance in learning the

inherent signatures of appliances as demonstrated in [141]. We further adjusted the PSPNet model for the NILM task, and the training structure of the adjusted PSPNet is shown in Figure 5.7.

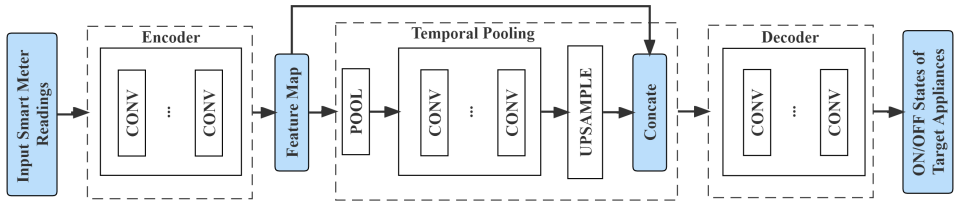


Figure 5.7: The overall layout of the deep learning model for NILM.

We provide a detailed description of the adjusted PSPNet model, which consists of three modules: the encoder, the temporal pooling module, and the decoder.

- **Encoder.** The input of the encoder is the household aggregated load consumption of the target appliances over a 1-hour interval (the consumption datasets were resampled to 30 seconds). The encoder makes up of four modules, each of which is alternated by a max pool layer except for the last block. The encoder increases the output features from a single aggregation value to 256 while paying the price of decreasing the time signal resolution by ten times.

- **Temporal Pooling.** The temporal pooling consists of four average pooling modules, filter sizes of which are decreased from the whole size of the input signal to one-sixth of it. After going through a convolutional layer, the feature dimension of the input is reduced to a quarter of its original size, and the acquired feature maps are upsampled to the size of the input time signals. Then the upsampled feature maps (shallow features) are concatenated with the original input signal (deep features) from the temporal pooling to get the final feature maps. The fusion of the deep and shallow features of the temporal pool could enable this block to get contextual information fed into the decoder.

- **Decoder.** The decoder receives the output from the temporal pooling block and passes it to a convolutional layer to recover the time signal resolution. Then the output is fed into the final convolutional layer to produce the final appliance-level load disaggregation.

5.5.2 Utility Optimization of DP^2-NILM

Recall the FL optimization objective (equation 5.6), the DP^2-NILM framework considers two utility optimization schemes, the FedAvg-NILM and the FedProx-NILM, to achieve this goal.

5.5.2.1 FedAvg-NILM

Algorithm 3 depicts the steps of FedAvg-NILM. The FedAvg [143] allows the smart

Algorithm 3: FedAvg-NILM

1 **Central Server Execution:**
2 Initialise the global model parameters w_G
3 **for** *each global round* $r \leq R$ **do**
4 | **for** *each client* $n \in \{1, 2, \ldots, N\}$ ***in parallel*** **do**
5 | | $w^n_{r+1} \leftarrow HouseholdsUpdate(w^n_r)$
6 | **end**
7 | $w_{r+1} \leftarrow \frac{\sum_{n=1}^{N} w^n_r}{N}$
8 **end**
9 Broadcast the global model to all clients
10 **Smart Meter Client Execution:**
11 **procedure** $HouseholdsUpdate(w^n_r)$:
12 Split L_n into batches of size B_L;
13 **for** *each local client epoch* $e \leq E$ **do**
14 | **for** *each batch of* L_n **do**
15 | | $w^n \leftarrow w^n - \eta \nabla \mathcal{L}(w^n)$
16 | **end**
17 **end**
18 Upload w^n to the central server

meter clients to train their local DNN models iteratively using the same learning rate and the number of epochs before uploading the updated model weights to the central server. For each global round (line 3), every smart meter client receives a copy of the global model and trains its local DNN models with its private smart meter readings for multiple epochs using $w^n \leftarrow w^n - \eta \nabla \mathcal{L}(w^n)$ (line 13-17), where η is the learning rate. After this, the local clients upload their updated local model weights w^n to the central server (line 18). Then, the central server updates the global model by averaging the uploaded weights from the smart meter clients (line 7) and broadcasts the updated global model to all clients (line 9).

An advantage of FedAvg-NILM is that a well-trained FedAvg-NILM model can outperform a single local NILM model while maintaining data privacy. Moreover, FedAvg has been proven to be efficient in reducing the communication overhead between the local clients and the global server [143].

Nevertheless, the FedAvg only performs effectively under the premise that all the local clients utilize a similar initialization. It has been shown that data heterogeneity impedes the convergence of FedAvg [118]. On the other hand, in real-world NILM tasks, smart meter clients often exhibit diverse appliance usage patterns, making the local client models easy to deviate from the global model, thereby reducing the overall performance.

5.5.2.2 FedProx-NILM

Data from smart meters are likely heterogeneous since they are collected under various contexts (e.g., across different countries) and are affected by diverse client behaviours leading to heterogeneous load usage distributions. Our DP^2-NILM framework is efficient for guaranteeing the convergence of the FL model in heterogeneity settings, i.e., the non-IID data settings, by incorporating FedProx [118] as an extension of the utility optimization scheme.

Algorithm 4: FedProx$_{mod}$-NILM

1 **Central Server Execution:**
2 (// Same central server execution steps as the FedAvg-NILM)
3 Broadcast the global model to all clients
4 **Smart Meter Client Execution:**
5 **procedure** $HouseholdsUpdate(w_r^n)$:
6 Split L_n into batches of size B_L
7 **for** *each local client epoch* $e \leq E$ **do**
8 **for** *each batch of* L_n **do**
9 $\nabla \mathcal{L}_{prox}(w^n) \leftarrow \nabla \mathcal{L}(w^n) + \mu(w^n - w_r^n)$
10 $w^n \leftarrow w^n - \eta \nabla \mathcal{L}_{prox}(w^n)$
11 **end**
12 **end**
13 Upload w^n to the central server

Algorithm 4 depicts the steps of FedProx-NILM. The central server executes the same steps as in the FedAvg-NILM. However, a proximal term $\mu(w^n - w_r^n)$ is added to update the local model of smart meter clients (line 9), which keeps local updates from deviating too much from the initial global model. When $\mu = 0$, the FedProx-NILM will produce the same results as the FedAvg-NILM.

Specifically, in the typical FedProx training paradigm, there is an inexact minimizer adjusting the local epoch of each client to reduce the negative impact of the system heterogeneous, which is defined as follows.

Definition 2 (γ-inexact Solution [118])**.** The w^* is a γ-inexact minimizer solution for the optimization objective in equation 5.6 if $\|w^* - w_r^n\| \leq \gamma \left\|w_r^n - w_{r-1}^n\right\|$, where $\gamma \in [0, 1)$.

The γ-inexact minimizer solution considers adjusting the local computation and the global communication overhead based on the number of local model epochs performed by the clients. In our framework, we hypothesize that most smart meter clients are available and capable of completing a certain number of local epochs. In contrast, for the very few stragglers, their destabilized training environment may produce models that contribute little to the FL global model. Therefore, we adjusted the FedProx to make it more efficient in the DP^2-NILM framework by utilising the proximal term $\mu(w^n - w_r^n)$ with the exact minimizer solution w_r^n rather than the inexact one.

5.5.3 Privacy-preserving of DP^2-NILM

The DP^2-NILM considers privacy-preserving mechanisms at two different levels to suit various privacy requirements from smart meter clients, i.e., the global differential privacy federated learning and the local differential privacy federated learning.

5.5.3.1 Global differential privacy federated learning NILM

In the DP^2-NILM paradigm, if a client sends out the privacy requirement and trusts the central server, the GDPFL-NILM will be utilized for this client. Although there must be a certain degree of trust in the central server, this presumption is significantly less stringent than granting the server access to the data. Algorithm 5 details the GDPFL-NILM scheme in the DP^2-NILM.

Algorithm 5: GDPFL-NILM

Central Server Execution:
Initialise the global model parameters w_G
for *each global round* $r \leq R$ **do**
  Compute privacy cost: $\hat{\epsilon}_r \leftarrow PrivacyAccount(\delta, \sigma)$;
  if $\hat{\epsilon}_r > \epsilon_r$ **then**
    return w_r
  end
  else
    for *each client* $n \in \{1, 2, \ldots, N\}$ ***in parallel*** **do**
      $w_{r+1}^n \leftarrow HouseholdsUpdate(w_r^n)$
    end
    $w_{r+1} \leftarrow \frac{\sum_{n=1}^{N} w_r^n}{N} + \mathcal{N}(0, \Delta\mathcal{F}^2 \cdot \sigma^2)$
  end
end
Broadcast the global model to all clients
Smart Meter Client Execution:
(// Same smart meter client execution steps as the FedAvg-NILM)
Upload w^n to the central server

Algorithm 6: LDPFL-NILM

1 **Central Server Execution:**
2 (// Same central server execution steps as the FedAvg-NILM)
3 Broadcast the global model to all clients
4 **Smart Meter Client Execution:**
5 **procedure** $HouseholdsUpdate(w_r^n)$:
6 Split L_n into batches of size B_L
7 **for** *each local client epoch* $e \leq E$ **do**
8 **for** *each batch of* L_n **do**
9 $\nabla\mathcal{L}_{ldp}(w^n) \leftarrow \nabla\mathcal{L}(w^n) + \mathcal{N}(0, \frac{\Delta\mathcal{F}^2 \cdot \sigma^2}{N})$
10 $w^n \leftarrow w^n - \eta\nabla\mathcal{L}_{ldp}(w^n)$
11 **end**
12 **end**
13 Upload w^n to the central server

In GDPFL-NILM, the smart meter client execution steps are the same as in FedAvg-NILM. The central server guarantees participant-level privacy by perturbing the model weights aggregation, i.e., adding Gaussian noise $\mathcal{N}(0, \Delta\mathcal{F}^2 \cdot \sigma^2)$ to the aggregated results (line 12). Moreover, to ensure the (ϵ, δ)-GDP, after each global round, the algorithm $PrivacyAccount()$ calculates the accumulated privacy budget (line 4). The global training iteration will be stopped if it exceeds the overall budget ϵ (line 6).

5.5.3.2 Local differential privacy federated learning NILM

In the LDPFL [4], smart meter clients apply noise on the updated local model weights before uploading them to the central server. The LDPFL-NILM scheme in DP^2-NILM is presented in Algorithm 6.

The central server updating process in the LDPFL-NILM is the same as in FedAvg-NILM. On the other hand, the smart meter clients guarantee their privacy by perturbing the updated local model weights, i.e., adding Gaussian noise $\mathcal{N}(0, \frac{\Delta\mathcal{F}^2 \cdot \sigma^2}{N})$ to the updated model weights (line 9). The LDPFL-NILM provides a better privacy notion than the GDPFL-NILM, and it is suitable for clients who require strict data privacy-preserving discipline.

5.5.4 Evaluation and Discussion

This section uses real-world smart meter datasets to evaluate the proposed DP^2-NILM framework. The datasets and the evaluation criteria are firstly introduced. Then, the performance of the FL setting in DP^2-NILM is compared with the Local-NILM models trained on individual household datasets and the Centralized-NILM

model trained on aggregated household datasets. After this, we examine the utility optimization schemes in the DP^2-NILM paradigm. Finally, based on the FedAvg, two privacy-preserving schemes, i.e., the GDPFL-NILM and the LDPFL-NILM, are compared in terms of the trade-off between model utility and privacy.

We used three real-world smart meter datasets to evaluate the DP^2-NILM framework, including UK-DALE [103], REDD [109] and REFIT [150]. As UK-DALE has been introduced in the subsection 3.3.4.2, Chapter 3, we here will introduce other two datasets.

- REDD: The Reference Energy Disaggregation Dataset (REDD) consists of six buildings in the U.S. from 3 to 19 days, and the sampling periods for mains and appliances are $1s$ and $6s$, respectively.
- REFIT: The REFIT dataset contains 20 buildings in the U.K. between 2013 and 2015 with an $8s$ sampling period for mains and appliances.

For comparison purposes, three appliances (fridge, dishwasher, and washing machine) are selected as our target appliances. 80% records from each smart meter client were selected as the training set, followed by a 10% for validation and 10% for testing. The distribution of the selected buildings is listed in Table 5.10.

Table 5.10: Distribution of the selected datasets. D: Days

Datasets	**Building**	**Period**	**Total (D)**	**Training (D)**	**Validation /Testing (D)**
UK-DALE	1	2013-04-12 to 2017-04-25	1475	1180	147.5
	2	2013-05-22 to 2013-10-03	135	108	13.5
	5	2014-06-29 to 2014-09-01	65	52	6.5
REDD	1	2011-04-19 to 2011-05-19	31	24.8	3.1
	2	2011-04-18 to 2011-05-21	34	27.2	3.4
	3	2011-04-17 to 2011-05-30	44	35.2	4.4
REFIT	2	2013-09-18 to 2015-05-27	617	493.6	61.7
	5	2013-09-27 to 2015-07-05	647	517.6	64.7
	9	2013-12-18 to 2015-07-07	567	453.6	56.7

TABLE 5.11 gives the relevant thresholds used in data preprocessing. We firstly filtered out the abnormal load consumption by the max power [102], and then down-sampled the load consumption of all the nine households from $6s$ to $30s$ through averaging. After this, the resampled data were normalized by subtracting the mean and dividing a constant load value 2000 W following Kelly and Knottenbelt [102]. Then the state series of each target appliance was derived from activation-time thresholding [102] as the input to feed into the DP^2-NILM.

Table 5.11: Relevant threshold information

	Fridge	**Dishwasher**	**Washing Machine**
Max power (W)	300	2500	2500
Power threshold (W)	50	20	20
Min. ON duration (s)	1	60	60
Min. OFF duration (s)	0	60	5

Furthermore, parameters used in the DP^2-NILM framework are listed in TABLE 5.12. We used TensorFlow to train the DP^2-NILM framework. To keep the comparison fair, all the models in our experiment use the same DNN architecture described in Section 5.7. For all the FL models in DP^2-NILM, each global training round consists of eight local epochs, allowing the clients to take reasonable learning steps before central server aggregation. For the privacy-preserving scheme, we vary the privacy budget ϵ between 4 and 12 while keeping $\delta = 10^{-5}$, and report the performance and attack success risk, i.e., the Accuracy and the ASR. The choice of $\delta = 10^{-5}$ satisfies the requirement that δ should be smaller than the inverse of the training data size [4]. Clipping is required to be a computationally efficient and common practice in deep learning to bound the sensitivity $\Delta\mathcal{F}^2$ of the gradients. With the TensorFlow Privacy framework, we implemented batch clipping with a threshold of 4. Further, for the listed parameters, we note that an additional parameter tuning step may improve the final model performance, however, at the cost of massive computational resources.

We use the same four evaluation metrics as in *FederatedNILM* to assess the model performance of the DP^2-NILM framework, i.e., the precision, recall, accuracy, and F_1 scores.

Moreover, to measure the privacy risk for the privacy-preserving DP^2-NILM models, we utilized the member inference attack metric defined by Yeom et al. [224]. To test the membership of an input record, this attack mechanism evaluates the loss of the uploaded local model parameters and then classifies it as a member if the loss is smaller than the average training loss. The attack success risk can be calculated as

$$ASR = TPR - FPR, \tag{5.8}$$

where $TPR = \frac{TP}{TP+FN}$ denotes the TP rate, and $FPR = \frac{FP}{FP+TN}$ represents the FP rate.

Table 5.12: Parameters used in the DP^2-NILM framework

Parameters	**Value**
Batch size	32
Global rounds	10
Local epochs	8
Number of clients	9
Proximal parameter μ	0.01
Privacy budget ϵ	[4, 8, 12]
Privacy relaxation term δ	10^{-5}
Gradient clipping threshold	4
Learning rate η	10^{-4}
Activation function	ReLU
Dropout probability	0.1
Momentum	0.5
Optimizer	SGD

5.5.4.1 Evaluation on the baseline model of DP^2-NILM

This subsection evaluates the FL setting of DP^2-NILM. There are three different model settings in this subsection:

- Local-NILM models: The Local-NILM models are trained on nine household datasets separately. This setting eliminates the need for data sharing with the central server but at the expense of updating all nine models separately.
- Centralized-NILM model: The Centralized-NILM model is trained on aggregated datasets from all nine households, which requires raw data sharing from the smart meter clients.
- FL-setting of DP^2-NILM (FedAvg-NILM): The FL-setting of DP^2-NILM utilizes FedAvg as the optimization method and trained on all the nine households without any exchange of the raw smart meter data. The FL model trained based on FedAvg in DP^2-NILM will be used later as the baseline for evaluating two schemes in the proposed framework.

For the Local-NILM models and the Centralized-NILM model, the epochs are set to 80 to achieve the final convergence. For comparison purposes, Table 5.13 lists the average performance scores of the Local-NILM models, the Centralized-NILM model, and the FL-setting of DP^2-NILM (FedAvg-NILM).

Table 5.13: Average performance scores of the Local-NILM models, the Centralized-NILM model, and the FL-setting of DP^2-NILM for 9 households

	Fridge				**Dishwasher**				**Washing Machine**			
	Accuracy	F_1	Precision	Recall	Accuracy	F_1	Precision	Recall	Accuracy	F_1	Precision	Recall
Local-NILM	0.83	0.79	0.82	0.74	0.98	0.88	0.86	0.83	0.99	0.78	0.89	0.70
Centralized-NILM	0.86	0.80	0.79	0.81	0.97	0.70	0.87	0.59	0.97	0.66	0.71	0.62
FedAvg-NILM	0.65	0.63	0.50	0.85	0.97	0.75	0.92	0.64	0.98	0.71	0.83	0.62

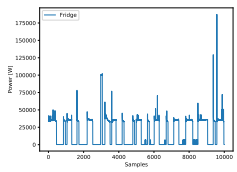
A comparison of FedAvg-NILM with Local-NILM models examines the performance of federated learning strategies in capturing diversities among clients. Moreover, comparing FedAvg-NILM to the Centralized-NILM model evaluates the overall performance of the common utility FL model. From the results, we can see that for each appliance, all models achieved satisfactory results on the dishwasher and washing machine and reasonable results on the fridge. Note that the FedAvg-NILM achieved the same accuracy score and higher F_1, precision, and recall scores on dishwashers and washing machines compared with the centralized-NILM model. For the fridge, as it consumes relatively low power compared with other appliances, it is likely to be learned with less evident signature during model training, and such consumption can easily be omitted as unidentified load noise in the FL paradigm. Overall, we can conclude that the FedAvg-NILM model in the DP^2-NILM framework works well, and its performance can be used as the baseline for further evaluations.

We also observed that the FedAvg-NILM might achieve more satisfying performances with more global rounds. For example, we have set 100 global rounds for the FedAvg-NILM. The final average accuracy for the fridge, dishwasher, and washing machine were 0.89, 0.99, and 0.99, respectively, which are even better than the centralized NILM model. However, parameter tuning in FL remains challenging due to the distributed environment, and the associated computational overhead [99]. We further stress that fixed global and local training rounds enable efficient, fair, and comparative evaluations in our DP^2-NILM framework.

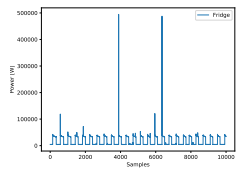
5.5.4.2 Evaluations on utility optimization of DP^2-NILM

Based on the evaluation results in the above subsection, although the performance of FedAvg-NILM in the case of 9 smart meter clients achieved satisfying performance on the dishwasher and washing machine, its scores on the fridge are worse than both the Local-NILM and the Centralize-NILM models. We further conjecture that the

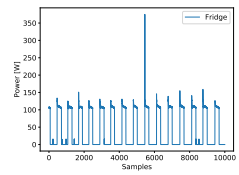
load consumption distribution of the fridge for the clients may be heterogeneous because they are collected geographically, i.e., the REDD dataset is from the U.S., whereas the other two datasets are from the U.K., and the size of the smart meter records from REDD are smaller than the other two datasets. Figure 5.8 shows the load consumption distribution of the fridge for all nine clients. It can be seen that different clients have different fridge usage patterns, and in particular, the UK-DALE and REFIT datasets differ significantly from those in REDD.



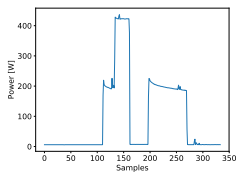
(a) UK-DALE House 1



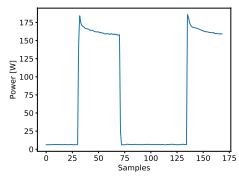
(b) UK-DALE House 2



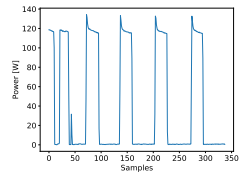
(c) UK-DALE House 5



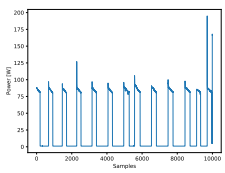
(d) REDD House 1



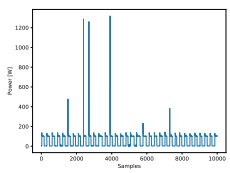
(e) REDD House 2



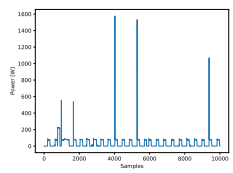
(f) REDD House 3



(g) REFIT House 2



(h) REFIT House 5



(i) REFIT House 9

Figure 5.8: Example fridge usage distributions for UK-DALE, REDD, and REFIT.

In this subsection, we will explore the relationship between data heterogeneity and the different types of FL utility optimising models. We hypothesize that using optimising algorithms that can accommodate statistical heterogeneity may help improve the performance of FL models. By comparing the FedProx-NILM to the FedAvg-NILM, we evaluate the ability of the two strategies to learn from heterogeneous data in the DP^2-NILM framework. Table 5.14 lists the average performance scores of the FedAvg-NILM and the FedProx-NILM, and we highlight the improve-

ment in blue and the downgrade in red of the FedProx-NILM corresponding to the FedAvg-NILM.

Table 5.14: Average performance scores of FedAvg-NILM and FedProx-NILM schemes for 9 households

	Fridge				**Dishwasher**				**Washing Machine**			
	Accuracy	F_1	Precision	Recall	Accuracy	F_1	Precision	Recall	Accuracy	F_1	Precision	Recall
FedAvg-NILM	0.65	0.63	0.50	0.85	0.97	0.75	0.92	0.64	0.98	0.71	0.83	0.62
FedProx-NILM	0.85	0.81	0.82	0.81	0.98	0.80	0.78	0.82	0.97	0.54	0.83	0.40
Evaluation	(↑ 20%)	(↑ 18%)	(↑ 32%)	(↓ 4%)	(↑ 1%)	(↑ 5%)	(↓ 19%)	(↑ 18%)	(↓ 1%)	(↓ 17%)	(-)	(↓ 22%)

It can be observed that FedProx-NILM significantly outperforms FedAvg for most scores on fridge and dishwasher, especially on the fridge, with an improved accuracy by 20% and an increased precision by 32%. On the other hand, there is a slight drop (1%) in accuracy and a significant drop (22%) in recall of the washing machine. We infer that, as the signatures of the washing machines are more complex than the other two appliances [57], the proximal term in FedProx-NILM reduces the difference in weight updates for individual models, which may undermine the learning of significant features of washing machines by the client models. To conclude, the above results confirm our assumptions regarding the utility optimization based on FedProx-NILM for handling heterogeneous smart meter appliances.

5.5.4.3 Evaluations on privacy-preserving of DP^2-NILM

FedAvg-NILM and FedProx-NILM presented unique advantages for devices with different signatures and datasets with different degrees of consistency. However, studies suggest that potential risks still exist in the training communication process even though the transmitted objects are the updated parameters instead of the original data [28]. Therefore, it is necessary to provide stronger privacy guarantees to the FL-based NILM. In this subsection, we evaluate two privacy-preserving schemes of DP^2-NILM, i.e., the GDPFL-NILM and the LDPFL-NILM. When clients decide whether to participate in the DP^2-NILM paradigm for smart meter data analysis, our framework serves as a reference for quantifying the potential privacy loss based on the privacy budget ϵ. By comparing the benefits of participating in the framework, clients can make an informed decision on whether to join.

Table 5.15 compares the GDPFL-NILM and the LDPFL-NILM trained with varied privacy budget ϵ, in which we again use the FedAvg-NILM as the baseline model. Intuitively, the Gaussian random noise will slow the convergence of

the GDPFL-NILM and the LDPFL-NILM models while providing stronger privacy guarantees for the local clients, leading to trade-off problems between model utility and privacy. It is noticed that even with a stronger privacy guarantee, the GDPFL-NILM still performs reasonably with a marginal reduction in accuracy score from 0.65 to 0.58 with the privacy budget $\epsilon = 8$. Interestingly, most model performance scores for the dishwasher and washing machine drop dramatically. This is likely because they differ from the fridge in terms of features, as dishwashers and washing machines may offer more insight into individual behaviour because they are more closely related to the routines of smart meter clients.

Table 5.15: Average performance scores of the GDPFL-NILM and the LDPFL-NILM schemes for 9 households

	Privacy	**Fridge**				**Dishwasher**				**Washing Machine**				**Privacy**	**Trusted**
	Budget	Accuracy	F_1	Precision	Recall	Accuracy	F_1	Precision	Recall	Accuracy	F_1	Precision	Recall	**Guarantee**	**Server**
FedAvg-NILM	\	0.65	0.63	0.50	0.85	0.97	0.75	0.92	0.64	0.98	0.71	0.83	0.62	Basic	Yes
GDPFL-NILM	4	0.54	0.53	0.37	0.95	0.90	0.14	0.16	0.72	0.95	0.68	0.40	0.92	Moderate	Yes
	8	0.63	0.61	0.49	0.84	0.97	0.69	0.93	0.56	0.98	0.68	0.79	0.60		
	12	0.66	0.82	0.81	0.83	0.99	0.85	0.86	0.85	0.98	0.74	0.80	0.63		
LDPFL-NILM	4	0.58	0.40	0.40	0.38	0.93	0.11	0.21	0.39	0.94	0.10	0.11	0.34	Strong	No
	8	0.58	0.42	0.41	0.44	0.94	0.20	0.30	0.40	0.96	0.20	0.40	0.47		
	12	0.65	0.42	0.36	0.50	0.94	0.13	0.26	0.48	0.96	0.43	0.40	0.50		

We then compare the performance of the GDPFL-NILM and the LDPFL-NILM in terms of privacy attacks. To determine whether a client has participated in a training session, we use the attack success risk introduced in the above as the evaluation criterion. Figure 5.9 illustrates ASRs based on various epsilon budgets for FedAvg-NILM, GDPFL-NILM, and LDPFL-NILM, respectively.

We evaluate all three models with three privacy budget values (i.e., $\epsilon = 4$, $\epsilon = 8$, and $\epsilon = 12$) with a fixed $\delta = 10^{-5}$. Figure 5.9 shows that LDPFL-NILM with the setting $\epsilon = 4$ mitigates the attack success risk better (downgrades the risk to 0.33) with compromises in decreasing model accuracy by 7% for fridge, 4% for dishwasher, and 4% for washing machine. The GDPFL-NILM with $\epsilon = 8$ achieved satisfying performance on all three appliances and reduced the attack accuracy to 0.59.

Not surprisingly, the LDPFL-NILM imposes more noise than the GDPFL-NILM, which provides stronger privacy guarantees but less utility due to a higher amount of noise. It is worth noting that with a higher privacy budget $\epsilon = 12$, the attack success risk in both the GDPFL-NILM and the LDPFL-NILM are similar to that in FedAvg-NILM whereas the F_1, precision, and recall for the LDPFL-NILM

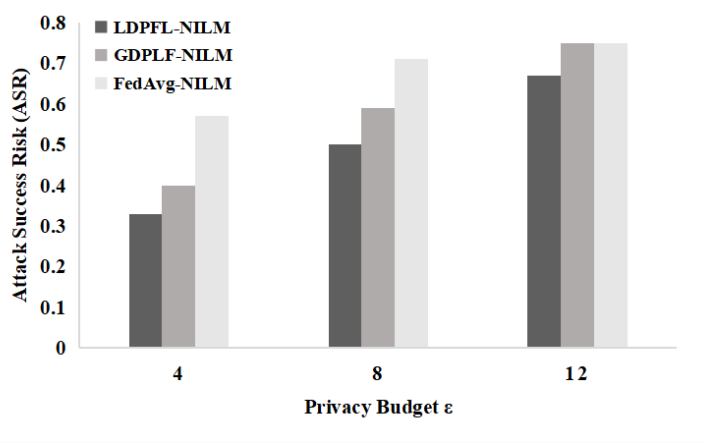


Figure 5.9: The ASRs of FedAvg-NILM, the GDPFL-NILM, and the LDPFL-NILM in the DP^2-NILM framework.

are much worse than the FedAvg-NILM and the GDPFL-NILM. Therefore, utilising the GDPFL-NILM or the FedAvg-NILM may achieve a better trade-off between utility and privacy when clients have a higher privacy budget.

5.6 Chapter Summary

In this chapter, we firstly proposed a distributed framework based on federated learning for NILM (FederatedNILM) to classify the typical states of appliances on the household level. The proposed FederatedNILM combines federated learning with a novel deep neural network model, which could benefit from labelled data of multiple distributed user data sources for NILM. Moreover, compared with the standard centralized model, the framework only requires model parameters uploading instead of data transmitting, which has good capability for data privacy protection. Comparative experiments on a real-world dataset demonstrate the feasibility and good performance of the proposed FederatedNILM framework and the superior model performance of the adopted deep learning architecture for NILM.

To provide privacy guarantees and optimize the model utility for the FederatedNILM, we proposed the DP^2-NILM framework based on federated learning and differential privacy for NILM, which provides two schemes to the smart meter clients, i.e., the utility optimization scheme and the privacy-preserving scheme. The utility optimization scheme consists of the FedAvg-NILM and the FedProx-NILM focusing on dealing with the data heterogeneity. The privacy-preserving scheme includes

the global differential privacy federated learning NILM and the local differential privacy federated learning NILM to provide privacy guarantees from various levels. We conducted extensive evaluations for the proposed DP^2-NILM framework based on real-world smart meter datasets and demonstrated its scalability from multiple perspectives.

The proposed DP^2-NILM framework will serve as a key technology to enable electric utilities and service providers to offer various large-scale smart energy services at the local/residential level, thereby enhancing the return on investment from the smart meters. Meanwhile, this framework enables smart meter clients to receive more valuable feedback, facilitating behaviour change in energy use and therefore contributing to the decarbonization of the energy system.

Chapter 6

Conclusion and Future Directions

6.1 Summary of Contributions

The following contributions to developing a distributed and real-time machine learning framework for smart meter big data are discussed in Chapter 3-5.

6.1.1 Load Profiling and Forecasting System for Multi-Level Smart Meter Data Analysis

In this chapter, we developed load profiling and forecasting frameworks for multi-Level smart meter data analysis in traditional offline settings, i.e., a model trained on the observed historical data. The contribution of this chapter can be summarized as follows:

- For the appliance level smart meter data analysis, we optimize the structure of a deep neural network based on sequence-to-point and transfer learning to improve the training efficiency of the model while ensuring the accuracy of NILM. Useful data analysis tools such as PCA and k-means clustering are used based on the disaggregation results of individual household appliances to infer the daily routine patterns of the smart meter users. The proposed appliance level smart meter data analysis framework can be used to address a wide variety of applications. For example, the proposed model can monitor the conditions of patients in real-time, and allows for early detection and timely intervention of the disease in the remote healthcare service.

- For the individual and higher aggregation level smart meter analysis, the proposed system uses consensus-based clustering to reduce the complexity of the building energy management process. A dynamic clustering structure is adopted in the proposed system, which can help to identify different load usage behaviours at different seasons. Moreover, personalized forecasting models for different cluster groups are developed by consensus-based model training strategy and explore the relationship between the performance of forecast algorithms and different consumption patterns at both the building and aggregation levels. The proposed system provides a systematic way to help to deliver customized load forecasting models, and enable robust, cost-effective building energy management. Moreover, the proposed overall approach has great potential to optimize the demand side management strategy by developing customized electricity dynamic pricing strategies and to help reduce greenhouse gas emissions and decrease non-renewable fuel reliance

6.1.2 From Offline to Online: Real-time Smart Meter Data Analysis

In this chapter, we consider the former load profiling and forecasting framework in the online setting, i.e., to analyze real-time smart meter big data. The main contributions of this chapter can be summarized as follows:

- We propose a two-unit universal online functional analysis model (Universal-OFA) with universal applicability for analysing load at different scales, such as individual and regional/district levels. To the best of our knowledge, no previous work has considered such a structure to investigate multi-scale load analysing problems.

- In the Universal-OFA, a loose sketch-detail refinement strategy is proposed to first sketch the loose features of load profiles and social information by clustering and training historical daily load curves at the individual level and then refine the details of clustered/trained models to generalize them into higher levels based on real-time load records.

- To assist the clustering process in the multi-scale load dynamic profiling unit, this work proposes an intra-day volatility score to capture more volatile features of the intra-day load curve. Furthermore, social data from participants are analysed to understand the clustered load profiles further.

- The proposed Universal-OFA reports the dynamic changes in the load usage patterns of different levels of participants. Furthermore, it can accurately

forecast newly joined customers without retraining the previous model, which can result in promising computational benefits, particularly for large and complex regions/districts where participants are likely to participate in the smart meter data analysis scheme asynchronously.

6.1.3 Distributed and Privacy-preserving Machine Learning Framework for Smart Meter Data Applications

Although federated learning, as a newly proposed distributed and privacy-preserving framework, has been studied in various areas, more attention should be paid to smart grid applications, especially for NILM. This chapter develops a distributed and privacy-preserving framework for appliance level smart meter big data analysis. The main contributions of this chapter are summarised as follows.

- We firstly develop the FederatedNILM, a distributed framework for NILM based on federated deep learning, to enable appliance level smart meter data analysis in the distributed environment. Specifically, we combined an advanced deep neural network architecture with federated learning, which could benefit the whole framework by providing more accurate state inference for multiple appliances on the household level. The comparative experiments verified the effectiveness of the proposed FederatedNILM by investigating key practical characteristics, including communication costs and model accuracy.

- We propose DP^2-NILM, an enhanced distributed and privacy-preserving framework based on FederatedNILM, which deploys federated learning with two enhancing schemes, i.e., the utility optimization and the privacy-preserving in real-world NILM scenarios. The DP^2-NILM is expected to provide a systematic understanding of different enhancement schemes for the challenges in real-world FL-based NILM applications to satisfy the diverse requirements of smart meter clients.

- To deal with client heterogeneity, the DP^2-NILM framework examines two utility optimization schemes derived from the FL paradigm, the Federated averaging (FedAvg) [143] and FedProx [118]. By exploring how the utility optimization schemes impact the accuracy of the FL-based NILM models, the proposed DP^2-NILM can achieve satisfactory performance under both homogeneous and heterogeneous data environments.

- The DP^2-NILM framework preserves data privacy by incorporating differential privacy (DP) at different levels, i.e., the central DP level and local DP level,

and aims to find a better trade-off between utility and privacy for various privacy requirements.

- We examine the correlations between key parameters and the inference accuracy of different enhancement schemes. It has been demonstrated that the proposed DP^2-NILM can be scalable and offers enlightening insights into different smart meter client requirements based on the extensive evaluation of three real-world datasets.

6.2 Future Directions

Based on the works done in this thesis, distributed and real-time machine learning frameworks for multi-level smart meter data analysis have been developed. Although the thesis fulfills the objectives of the project, there are still several directions that can be explored in the future.

For the appliance level smart meter analysis, anomaly detection techniques for the proposed ADLs pattern recognition NILM model in Chapter 3 can be combined. In this way, based on the ADLs pattern recognition analysis, the model should consistently monitor the real-time daily activities of the smart users under the protocols of privacy protection and alert the abnormal behaviours in time. Moreover, based on the FederatedNILM and DP^2-NILM proposed in Chapter 5, we stress that the framework has the potential to accommodate a wider range of client types, such as commercial and industrial clients. Further, as proved by McMahan et al. [144], the federated learning scheme has the potential to perform better in the later stage of convergence by reducing number of the local epochs and learning rates, one of our potential future direction is to further explore the convergence property of the federated learning-based NILM model. Nevertheless, the training environment for such client types may be more complex, so it is important to further consider the system heterogeneity to ensure the robustness of the framework. Furthermore, as smart devices enable real-time feedback from smart meter clients, adapting the DP^2-NILM framework to online scenarios will deliver more flexible smart meter data analysis and improve the communication efficiency of the FL paradigm.

For the individual level and the higher aggregation level smart meter data analysis, a future direction worth exploring for the Universal-OFA proposed in Chapter 4 would be to devise a method by which the Universal-OFA can detect pattern shifts between weekends and weekdays promptly since it has been observed that the Universal-OFA has difficulty capturing the pattern shifts in electricity consumption behaviour between the weekdays and weekends. Further, even though

the loose sketch layer of the Universal-OFA does not require participants to provide their identity information, the detail refinement layer requires real-time load records of individuals. Therefore, advanced privacy-preserving algorithms should be provided for the detailed refinement layer, such as federated learning [127, 40, 38]. Furthermore, the current Universal-OFA takes into account the residential sector. As the Universal-OFA framework can be deployed at various data scales, we will examine its scalability in various real-world industry scenarios in the near future. At the aggregation level, the forecast performance of different learning algorithms is analysed in Chapter 3 and Chapter 4. It has been shown that aggregation reduces the randomness and variations of individual buildings and produces more stable forecast performance among different models. Moreover, the experiment results in Chapter 3 indicate that some important features of individual buildings are lost in the data aggregation process. As a result, it is necessary to develop a better data aggregation process to retain more critical features and further improve forecast performance for the aggregation level smart meter analysis.

In this thesis, we have considered TL, OL, and FL in the three core chapters, which are three enhancing approaches built on standard machine learning techniques to address practical challenges in smart meter big data analysis. Extending these techniques to a comprehensive framework, i.e., online FTL, will enable the developing of an advanced machine learning framework for smart meter big data analysis. Fig. 6.1 illustrates the proposed online FTL framework that is described below. The data in the source domain can be generated in real-time or from pregiven datasets. It should be noted that a scratch of the source data is essential to ensure the benchmark performance of the source models. Each local device in the target domain generates data in an online fashion, and the real-time data is analysed by online learners, who then attempt to formulate an optimal strategy for online updating during each training round [87]. The global model enables model aggregation, heterogeneous computing, updating, and broadcasting. Local devices, such as smartphones and laptops, provide essential infrastructure tools, including local online/offline training, uploading, and distributed storage.

Various applications may be developed on top of the proposed online FTL to provide critical human-machine interface services. By utilising federated learning, machine learning models for multiple parties can be established without exporting local data, ensuring data security and privacy while providing users with tailored services. Meanwhile, TL enables FL to train models on a variety of different but related parties, which is practically important given that stakeholders within the same FL framework are usually from the same sector. Furthermore, classical batch/offline learning has low efficiency in computing costs and limited scalability for large-

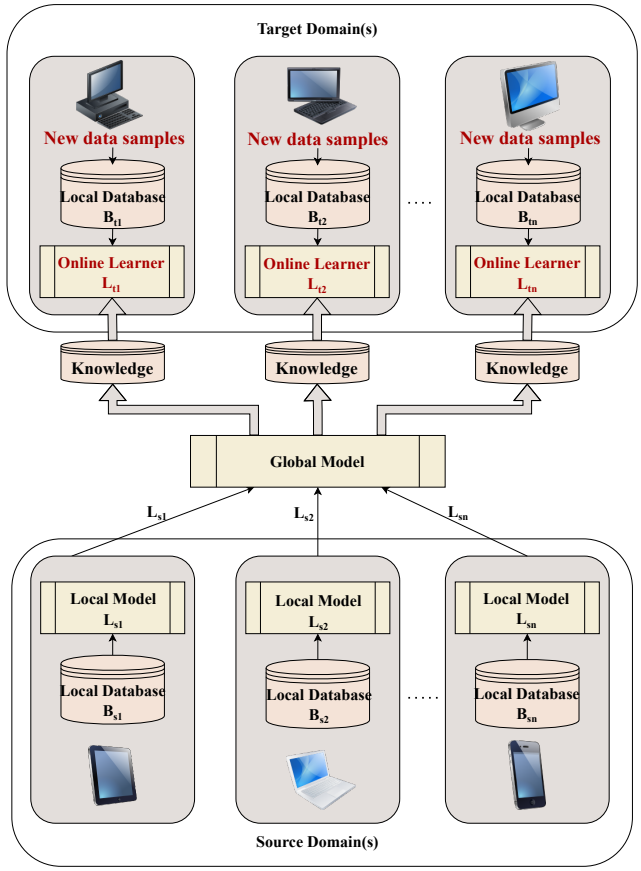


Figure 6.1: A vision of online FTL framework

scale applications due to the need for model retraining after online data sequences are generated. We envisage that extending TL, OL, and FL to online FTL will help overcome the limitations of traditional batch learning by allowing online learners to update the local model safely and rapidly.

Bibliography

[1] Cardiff council carbonculture. `https://platform.carbonculture.net/communities/cardiff-council/19/`, 2019. accessed: 06 November 2019.

[2] Office of gas and electricity markets (ofgem), transition to smart meters. https://www.ofgem.gov.uk/gas/retail-market/metering/transition-smart-meters/, 2020. [accessed 18 December 2020].

[3] M. S. H. Abad, E. Ozfatura, D. GUndUz, and O. Ercetin. Hierarchical federated learning across heterogeneous cellular networks. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8866–8870, 2020. doi: 10.1109/ICASSP40776.2020.9054634.

[4] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[5] Ali Adabi, Patrick Mantey, Emil Holmegaard, and Mikkel Baun Kjaergaard. Status and challenges of residential and industrial non-intrusive load monitoring. In *2015 IEEE Conference on Technologies for Sustainability (SusTech)*, pages 181–188. IEEE, 2015.

[6] S. Agatonovic-Kustrin and R. Beresford. Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research. *Journal of Pharmaceutical & Biomedical Analysis*, 22:717–727, 2000.

[7] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. Time-series clustering–a decade review. *Information systems*, 53:16–38, 2015.

[8] Muhammad Waseem Ahmad, Monjur Mourshed, and Yacine Rezgui. Tree-based ensemble methods for predicting pv power generation and their comparison with support vector regression. *Energy*, 164:465–474, 2018.

[9] Tanveer Ahmad and Huanxin Chen. Potential of three variant machine-learning models for forecasting district level medium-term and long-term energy demand in smart grid environment. *Energy*, 160:1008–1020, 2018.

[10] Nesreen K Ahmed, Amir F Atiya, Neamat El Gayar, and Hisham El-Shishiny. An empirical comparison of machine learning models for time series forecasting. *Econometric reviews*, 29(5-6):594–621, 2010.

[11] José M Alcalá, Jesús Ureña, Álvaro Hernández, and David Gualda. Assessing human activity in elderly people using non-intrusive load monitoring. *Sensors*, 17(2):351, 2017.

[12] Nima Amjady. Short-term hourly load forecasting using time-series modeling with peak load estimation capability. *IEEE Transactions on power systems*, 16(3):498–505, 2001.

[13] Rebecca R Andridge and Roderick JA Little. A review of hot deck imputation for survey non-response. *International statistical review*, 78(1):40–64, 2010.

[14] Irish Social Science Data Archive. Data from the commission for energy regulation (cer) – smart metering project., August 2012. URL `http://www.ucd.ie/issda/data/commissionforenergyregulationcer/`.

[15] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[16] Mayur Barman, NB Dev Choudhury, and Suman Sutradhar. A regional hybrid goa-svm model based on similar day approach for short-term load forecasting in assam, india. *Energy*, 145:710–720, 2018.

[17] Peter Bartlett, Elad Hazan, and Alexander Rakhlin. Adaptive online gradient descent. In *Advances in Neural Information Processing Systems 20: Proceedings of the 2007 Conference*, pages 65–72. Neural Information Processing Systems (NIPS) Foundation, 2009.

[18] M Beiraghi and AM Ranjbar. Discrete fourier transform based approach to forecast monthly peak load. In *2011 Asia-Pacific Power and Energy Engineering Conference*, pages 1–5. IEEE, 2011.

[19] Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, page 359–370. AAAI Press, 1994.

[20] Gautam Bhattacharya, Koushik Ghosh, and Ananda S Chowdhury. Granger causality driven ahp for feature weighted knn. *Pattern Recognition*, 66:425–436, 2017.

[21] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[22] Salah Bouktif, Ali Fiaz, Ali Ouni, and Mohamed Adel Serhani. Optimal deep learning lstm model for electric load forecasting using feature selection and genetic algorithm: Comparison with machine learning approaches. *Energies*, 11(7):1636, 2018.

[23] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[24] Mengmeng Cai, Manisa Pipattanasomporn, and Saifur Rahman. Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques. *Applied energy*, 236:1078–1088, 2019.

[25] Luis M Candanedo, Véronique Feldheim, and Dominique Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and buildings*, 140:81–97, 2017.

[26] Hui Cao, Shubo Liu, Longfei Wu, Zhitao Guan, and Xiaojiang Du. Achieving differential privacy against non-intrusive load monitoring in smart grid: A fog computing approach. *Concurrency and Computation: Practice and Experience*, 31(22):e4528, 2019.

[27] 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.

[28] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 267–284, 2019.

[29] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[30] Nicolo Cesa-Bianchi, Alex Conconi, and Claudio Gentile. A second-order perceptron algorithm. *SIAM Journal on Computing*, 34(3):640–668, 2005.

[31] Mohamed Chaouch. Clustering-based improvement of nonparametric functional time series forecasting: Application to intra-day household-level load curves. *IEEE Transactions on Smart Grid*, 5(1):411–419, 2013.

[32] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

[33] Lilin Cheng, Haixiang Zang, Yan Xu, Zhinong Wei, and Guoqiang Sun. Probabilistic residential load forecasting based on micrometeorological data and customer consumption pattern. *IEEE Transactions on Power Systems*, 36(4): 3762–3775, 2021.

[34] Jeanno CL Cheung, Tat Wing Chim, Siu-Ming Yiu, Victor OK Li, and Lucas CK Hui. Credential-based privacy-preserving power request scheme for smart grid network. In *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011*, pages 1–5. IEEE, 2011.

[35] Carlos Coello and Efrén Mezura-Montes. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, 16:193–203, 07 2002. doi: 10.1016/S1474-0346(02)00011-3.

[36] Peter M Connor, Philip E Baker, Dimitrios Xenias, Nazmiye Balta-Ozkan, Colin J Axon, and Liana Cipcigan. Policy and regulation for smart grids in the united kingdom. *Renewable and Sustainable Energy Reviews*, 40:269–286, 2014.

[37] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[38] Shuang Dai and Fanlin Meng. Addressing modern and practical challenges in machine learning: A survey of online federated and transfer learning. *Applied Intelligence*, 2022. doi: 10.1007/s10489-022-04065-3.

[39] Shuang Dai, Fanlin Meng, Hongsheng Dai, Qian Wang, and Xizhong Chen. Electrical peak demand forecasting-a review. *arXiv preprint arXiv:2108.01393*, 2021.

[40] Shuang Dai, Fanlin Meng, Qian Wang, and Xizhong Chen. Dp ^{2}-nilm: A distributed and privacy-preserving framework for non-intrusive load monitoring. *arXiv preprint arXiv:2207.00041*, 2022.

[41] Wenyuan Dai, Qiang Yang, Gui rong Xue, and Yong Yu. Boosting for transfer learning. In *In ICML*, 2007.

[42] Yeming Dai, Xinyu Yang, and Mingming Leng. Forecasting power load: A hybrid forecasting method with intelligent data processing and optimized artificial intelligence. *Technological Forecasting and Social Change*, 182:121858, 2022.

[43] Manoranjan Dash and Huan Liu. Feature selection for classification. *Intelligent data analysis*, 1(1-4):131–156, 1997.

[44] Nguyen Quang Dat, Nguyen Thi Ngoc Anh, Nguyen Nhat Anh, and Vijender Kumar Solanki. Hybrid online model based multi seasonal decompose for short-term electricity load forecasting using arima and online rnn. *Journal of Intelligent & Fuzzy Systems*, 41(5):5639–5652, 2021.

[45] Erick Meira de Oliveira and Fernando Luiz Cyrino Oliveira. Forecasting mid-long term electric energy consumption through bagging arima and exponential smoothing methods. *Energy*, 144:776–788, 2018.

[46] Maria Valentina Niño de Zepeda, Fanlin Meng, Jinya Su, Xiao-Jun Zeng, and Qian Wang. Dynamic clustering analysis for driving styles identification. *Engineering Applications of Artificial Intelligence*, 97:104096, 2021.

[47] Ofer Dekel, Shai Shalev-Shwartz, and Yoram Singer. The forgetron: A kernel-based perceptron on a budget. *SIAM Journal on Computing*, 37(5):1342–1372, 2008.

[48] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13(1), 2012.

[49] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[50] Xiang Deng, Zijing Tan, Meijing Tan, and Wenjing Chen. A clustering-based climatic zoning method for office buildings in china. *Journal of Building Engineering*, 42:102778, 2021.

[51] Jagjeet Dhillon, Shah Atiqur Rahman, Sabbir U Ahmad, and Md Jahangir Hossain. Peak electricity load forecasting using online support vector regression. In *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–4. IEEE, 2016.

[52] 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.

[53] Bing Dong, Cheng Cao, and Siew Eang Lee. Applying support vector machines to predict building energy consumption in tropical region. *Energy and Buildings*, 37(5):545–553, 2005.

[54] Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *Proceedings of the 25th international conference on Machine learning*, pages 264–271, 2008.

[55] Harris Drucker, Chris JC Burges, Linda Kaufman, Alex Smola, Vladimir Vapnik, et al. Support vector regression machines. *Advances in neural information processing systems*, 9:155–161, 1997.

[56] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4): 211–407, 2014.

[57] 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.

[58] Eric Eaton and Marie desJardins. Selective transfer between learning tasks using task-based boosting. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pages 337–342, 2011.

[59] Zekeriya Erkin, Juan Ramón Troncoso-Pastoriza, Reginald L Lagendijk, and Fernando Pérez-González. Privacy-preserving data aggregation in smart metering systems: An overview. *IEEE Signal Processing Magazine*, 30(2):75–86, 2013.

[60] Francisco Escolano, Pablo Suau, and Boyán Bonev. Feature selection and transformation. *Information Theory in Computer Vision and Pattern Recognition*, pages 211–269, 2009.

[61] Marcelo Espinoza, Caroline Joye, Ronnie Belmans, and Bart De Moor. Short-term load forecasting, profile identification, and customer segmentation: a methodology based on periodic time series. *IEEE Transactions on Power Systems*, 20(3):1622–1630, 2005.

[62] Seyedeh Narjes Fallah, Mehdi Ganjkhani, Shahaboddin Shamshirband, and Kwok-wing Chau. Computational intelligence on short-term load forecasting: A methodological overview. *Energies*, 12(3):393, 2019.

[63] Guo-Feng Fan, Yan-Hui Guo, Jia-Mei Zheng, and Wei-Chiang Hong. Application of the weighted k-nearest neighbor algorithm for short-term load forecasting. *Energies*, 12(5):916, 2019.

[64] Xin Fu, Xiao-Jun Zeng, Pengpeng Feng, and Xiuwen Cai. Clustering-based short-term load forecasting for residential electricity under the increasing-block pricing tariffs in china. *Energy*, 165:76–89, 2018.

[65] Santiago Gallón and Jorge Barrientos. Forecasting the colombian electricity spot price under a functional approach. *International Journal of Energy Economics and Policy*, 11(2):67–74, 2021.

[66] Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*. SIAM, 2020.

[67] Peng Gao, Weifei Wu, and Jingmei Li. Multi-source fast transfer learning algorithm based on support vector machine. *Applied Intelligence*, 51(11): 8451–8465, 2021.

[68] Everette S Gardner Jr. Exponential smoothing: The state of the art. *Journal of forecasting*, 4(1):1–28, 1985.

[69] Mohammad Ghomi, Mahdi Goodarzi, and Mahmood Goodarzi. Peak load forecasting of electric utilities for west province of iran by using neural network without weather information. In *2010 12th International Conference on Computer Modelling and Simulation*, pages 28–32. IEEE, 2010.

[70] Aldo Goia, Caterina May, and Gianluca Fusai. Functional clustering and linear regression for peak load forecasting. *International Journal of Forecasting*, 26:700 – 711, 2010.

[71] Aldo Goia, Caterina May, and Gianluca Fusai. Functional clustering and linear regression for peak load forecasting. *International Journal of Forecasting*, 26(4):700–711, 2010.

[72] Steven Golovkine, Nicolas Klutchnikoff, and Valentin Patilea. Clustering multivariate functional data using unsupervised binary trees. *Computational Statistics & Data Analysis*, 168:107376, 2022.

[73] HM Government. Carbon plan. `https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/47621/1358-the-carbon-plan.pdf`.

[74] Deniz Gündüz, Jesus Gomez-Vilardebo, Onur Tan, and H Vincent Poor. Information theoretic privacy for smart meters. In *2013 information theory and applications workshop (ITA)*, pages 1–7. IEEE, 2013.

[75] Stephen Haben, Colin Singleton, and Peter Grindrod. Analysis and clustering of residential customers energy behavioral demand using smart meter data. *IEEE transactions on smart grid*, 7(1):136–144, 2015.

[76] Stephen Haben, Colin Singleton, and Peter Grindrod. Analysis and clustering of residential customers energy behavioral demand using smart meter data. *IEEE transactions on smart grid*, 7(1):136–144, 2015.

[77] Stephen Haben, Siddharth Arora, Georgios Giasemidis, Marcus Voss, and Danica Vukadinović Greetham. Review of low voltage load forecasting: Methods, applications, and recommendations. *Applied Energy*, 304:117798, 2021.

[78] Te Han, Chao Liu, Wenguang Yang, and Dongxiang Jiang. Learning transferable features in deep convolutional neural networks for diagnosing unseen machine conditions. *ISA transactions*, 93:341–353, 2019.

[79] Clara Happ and Sonja Greven. Multivariate functional principal component analysis for data observed on different (dimensional) domains. *Journal of the American Statistical Association*, 113(522):649–659, 2018.

[80] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[81] George William Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.

[82] Muneeb Ul Hassan, Mubashir Husain Rehmani, and Jinjun Chen. Performance evaluation of differential privacy mechanisms in blockchain based smart metering. *arXiv preprint arXiv:2007.09802*, 2020.

[83] Xin He, Yushi Chen, and Pedram Ghamisi. Heterogeneous transfer learning for hyperspectral image classification based on convolutional neural network. *IEEE Transactions on Geoscience and Remote Sensing*, 58(5):3246–3263, 2019.

[84] Sepp Hochreiter and Jürgen Schmidhuber. Lstm can solve hard long time lag problems. *Advances in neural information processing systems*, 9, 1996.

[85] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[86] Steven CH Hoi, Jialei Wang, and Peilin Zhao. Libol: A library for online learning algorithms. *Journal of Machine Learning Research*, 15(1):495, 2014.

[87] Steven CH Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289, 2021.

[88] Sean D Holcomb, William K Porter, Shaun V Ault, Guifen Mao, and Jin Wang. Overview on deepmind and its alphago zero ai. In *Proceedings of the 2018 international conference on big data and education*, pages 67–71, 2018.

[89] Emil Holmegaard and Mikkel Baun Kjærgaard. Nilm in an industrial setting: A load characterization and algorithm evaluation. In *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 1–8. IEEE, 2016.

[90] Arne Holst. Number of internet of things (iot) connected devices worldwide from 2019 to 2030, Aug 2021. URL `https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/`.

[91] Che-Chiang Hsu and Chia-Yon Chen. Regional load forecasting in taiwan—applications of artificial neural networks. *Energy conversion and Management*, 44(12):1941–1949, 2003.

[92] David Hsu. Comparison of integrated clustering methods for accurate and stable prediction of building energy consumption data. *Applied energy*, 160: 153–163, 2015.

[93] Can Huang, Chih-Che Sun, Nan Duan, Yuming Jiang, Chloe Applegate, Peter D Barnes, and Emma Stewart. Smart meter pinging and reading through ami two-way communication networks to monitor grid edge devices and ders. *IEEE Transactions on Smart Grid*, 2021.

[94] Jakob Huber and Heiner Stuckenschmidt. Daily retail demand forecasting using machine learning with emphasis on calendric special days. *International Journal of Forecasting*, 36(4):1420–1438, 2020.

[95] Nathaniel Hudson, Md Jakir Hossain, Minoo Hosseinzadeh, Hana Khamfroush, Mahshid Rahnamay-Naeini, and Nasir Ghani. A framework for edge intelligent smart distribution grids via federated learning. In *2021 International Conference on Computer Communications and Networks (ICCCN)*, pages 1–9. IEEE, 2021.

[96] Rob J Hyndman and George Athanasopoulos. Forecasting: Principles and practice. *London: Bowker-Saur. Pharo*, 2014.

[97] Mayank Jain, Tarek AlSkaif, and Soumyabrata Dev. Validating clustering frameworks for electric load demand profiles. *IEEE Transactions on Industrial Informatics*, 17(12):8057–8065, 2021.

[98] Ali Javed, Byung Suk Lee, and Donna M Rizzo. A benchmark study on time series clustering. *Machine Learning with Applications*, 1:100001, 2020.

[99] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[100] Georgios Kalogridis, Costas Efthymiou, Stojan Z Denic, Tim A Lewis, and Rafael Cepeda. Privacy for smart meters: Towards undetectable appliance load signatures. In *2010 First IEEE International Conference on Smart Grid Communications*, pages 232–237. IEEE, 2010.

[101] Sidney Katz, Amasa B Ford, Roland W Moskowitz, Beverly A Jackson, and Marjorie W Jaffe. Studies of illness in the aged: the index of adl: a standardized measure of biological and psychosocial function. *Jama*, 185(12):914–919, 1963.

[102] 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.

[103] 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.

[104] I Kevin, Kai Wang, Xiaokang Zhou, Wei Liang, Zheng Yan, and Jinhua She. Federated transfer learning based cross-domain prediction for smart manufacturing. *IEEE Transactions on Industrial Informatics*, 2021.

[105] Hashir Moheed Kiani and Xiao-Jun Zeng. A function-on-function linear regression approach for short-term electric load forecasting. In *2019 IEEE Texas Power and Energy Conference (TPEC)*, pages 1–5. IEEE, 2019.

[106] Hashir Moheed Kiani and Xiao-Jun Zeng. A function-on-function linear regression approach for short-term electric load forecasting. In *2019 IEEE Texas Power and Energy Conference (TPEC)*, pages 1–5. IEEE, 2019.

[107] Hyungsul Kim, Manish Marwah, Martin Arlitt, Geoff Lyon, and Jiawei Han. Unsupervised disaggregation of low frequency power measurements. In *Proceedings of the 2011 SIAM international conference on data mining*, pages 747–758. SIAM, 2011.

[108] Jin-Gyeom Kim and Bowon Lee. Appliance classification by power signal analysis based on multi-feature combination multi-layer lstm. *Energies*, 12 (14):2804, 2019.

[109] 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.

[110] 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.

[111] Bo-Sung Kwon, Rae-Jun Park, and Kyung-Bin Song. Short-term load forecasting based on deep neural networks using lstm layer. *Journal of Electrical Engineering & Technology*, 15(4):1501–1509, 2020.

[112] Wouter Labeeuw, Jeroen Stragier, and Geert Deconinck. Potential of active demand reduction with residential wet appliances: A case study for belgium. *IEEE Transactions on Smart Grid*, 6(1):315–323, 2014.

[113] Ali Lahouar and J Ben Hadj Slama. Day-ahead load forecast using random forest and expert input selection. *Energy Conversion and Management*, 103: 1040–1051, 2015.

[114] Chr Lamnatou, D Chemisana, and C Cristofari. Smart grids and smart technologies in relation to photovoltaics, storage systems, buildings and the environment. *Renewable Energy*, 2021.

[115] Steve Lauriks, Annika Reinersmann, Henriëtte Geralde Van der Roest, FJM Meiland, Richard J Davies, Ferial Moelaert, Maurice D Mulvenna, Chris D Nugent, and Rose-Marie Dröes. Review of ict-based services for identified unmet needs in people with dementia. *Ageing research reviews*, 6(3):223–246, 2007.

[116] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: vision, hype and reality for data privacy and protection. *arXiv preprint arXiv:1907.09693*, 2019.

[117] Qiong Li, Qinglin Meng, Jiejin Cai, Hiroshi Yoshino, and Akashi Mochida. Applying support vector machine to predict hourly cooling load in the building. *Applied Energy*, 86(10):2249–2256, 2009.

[118] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[119] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2019.

[120] Yanying Li, Jinxing Che, and Youlong Yang. Subsampled support vector regression ensemble for short term electric load forecasting. *Energy*, 164:160–170, 2018.

[121] Zibin Li, Bo Liu, and Yanshan Xiao. Cluster and dynamic-tradaboost-based transfer learning for text classification. In *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pages 2291–2295. IEEE, 2017.

[122] Kai Liu, Hao Zhang, Joseph Kee-Yin Ng, Yusheng Xia, Liang Feng, Victor CS Lee, and Sang H Son. Toward low-overhead fingerprint-based indoor localization via transfer learning: Design, implementation, and evaluation. *IEEE Transactions on Industrial Informatics*, 14(3):898–908, 2017.

[123] Pengfei Liu, Xipeng Qiu, Xinchi Chen, Shiyu Wu, and Xuan-Jing Huang. Multi-timescale long short-term memory neural network for modelling sentences and documents. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2326–2335, 2015.

[124] Tie-Yan Liu, Wei Chen, and Taifeng Wang. Distributed machine learning: Foundations, trends, and practices. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 913–915, 2017.

[125] Xin Liu, Zijun Zhang, and Zhe Song. A comparative study of the data-driven day-ahead hourly provincial load forecasting methods: From classical data mining to deep learning. *Renewable and Sustainable Energy Reviews*, 119: 109632, 2020.

[126] Yang Liu, Yan Kang, Chaoping Xing, Tianjian Chen, and Qiang Yang. A secure federated transfer learning framework. *IEEE Intelligent Systems*, 35 (4):70–82, 2020.

[127] Yi Liu, JQ James, Jiawen Kang, Dusit Niyato, and Shuyu Zhang. Privacy-preserving traffic flow prediction: A federated learning approach. *IEEE Internet of Things Journal*, 7(8):7751–7763, 2020.

[128] Zixu Liu, Xiaojun Zeng, and Fanlin Meng. An integration mechanism between demand and supply side management of electricity markets. *Energies*, 11(12): 3314, 2018.

[129] Mingsheng Long, Yue Cao, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Transferable representation learning with deep adaptation networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(12):3071–3085, 2018.

[130] Miguel López, Sergio Valero, Ana Rodriguez, Iago Veiras, and Carolina Senabre. New online load forecasting system for the spanish transport system operator. *Electric Power Systems Research*, 154:401–412, 2018.

[131] Hongfang Lu, Feifei Cheng, Xin Ma, and Gang Hu. Short-term prediction of building energy consumption employing an improved extreme gradient boosting model: a case study of an intake tower. *Energy*, 203:117756, 2020.

[132] Haipeng Luo, Alekh Agarwal, Nicolo Cesa-Bianchi, and John Langford. Efficient second order online learning by sketching. *Advances in Neural Information Processing Systems*, 29:902–910, 2016.

[133] Jian Luo, Tao Hong, Zheming Gao, and Shu-Cherng Fang. A robust support vector regression model for electric load forecasting. *International Journal of Forecasting*, 2022.

[134] Lingling Lv, Zongyu Wu, Jinhua Zhang, Lei Zhang, Zhiyuan Tan, and Zhihong Tian. A vmd and lstm based hybrid model of load forecasting for power grid security. *IEEE Transactions on Industrial Informatics*, 18(9):6474–6482, 2021.

[135] Lennart Magnusson, Elizabeth Hanson, and Martin Borg. A literature review study of information and communication technology as a support for frail older people living at home and their family carers. *Technology and Disability*, 16 (4):223–235, 2004.

[136] Umer Majeed, Sheikh Salman Hassan, and Choong Seon Hong. Cross-silo model-based secure federated transfer learning for flow-based traffic classification. In *2021 International Conference on Information Networking (ICOIN)*, pages 588–593. IEEE, 2021.

[137] 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.

[138] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, Angela Hung Byers, et al. *Big data: The next frontier for innovation, competition, and productivity.* McKinsey Global Institute, 2011.

[139] Francisco Martínez-Álvarez, Amandine Schmutz, Gualberto Asencio-Cortés, and Julien Jacques. A novel hybrid algorithm to forecast functional time series based on pattern sequence similarity with application to electricity demand. *Energies*, 12(1):94, 2018.

[140] Pedro BM Martins, José GRC Gomes, Vagner B Nascimento, and Antonio R de Freitas. Application of a deep learning generative model to load disaggregation for industrial machinery power consumption monitoring. In *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–6. IEEE, 2018.

[141] Luca Massidda, Marino Marrocu, and Simone Manca. Non-intrusive load disaggregation by convolutional neural network and multilabel classification. *Applied Sciences*, 10(4):1454, 2020.

[142] Fintan McLoughlin, Aidan Duffy, and Michael Conlon. A clustering approach to domestic electricity load profile characterisation using smart metering data. *Applied energy*, 141:190–199, 2015.

[143] 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.

[144] 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.

[145] Manohar Mishra, Janmenjoy Nayak, Bighnaraj Naik, and Ajith Abraham. Deep learning in electrical utility industry: A comprehensive review of a decade of research. *Engineering Applications of Artificial Intelligence*, 96: 104000, 2020.

[146] Elena Mocanu, Phuong H Nguyen, Madeleine Gibescu, and Wil L Kling. Deep learning for estimating building energy consumption. *Sustainable Energy, Grids and Networks*, 6:91–99, 2016.

[147] Ramyar Rashed Mohassel, Alan Fung, Farah Mohammadi, and Kaamran Raahemifar. A survey on advanced metering infrastructure. *International Journal of Electrical Power & Energy Systems*, 63:473–484, 2014.

[148] Pablo Montero and José A. Vilar. Tsclust: An r package for time series clustering. *Journal of Statistical Software*, 62(1):1–43, 2014. doi: 10.18637/jss.v062.i01. URL `https://www.jstatsoft.org/index.php/jss/article/view/v062i01`.

[149] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640, 2021.

[150] 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.

[151] Shahzad Muzaffar and Afshin Afshari. Short-term load forecasts using lstm networks. *Energy Procedia*, 158:2922–2927, 2019.

[152] Mohammadreza Nazari, Afshin Oroojlooy, Lawrence Snyder, and Martin Takác. Reinforcement learning for solving the vehicle routing problem. *Advances in neural information processing systems*, 31, 2018.

[153] Bishnu Nepal, Motoi Yamaha, Aya Yokoe, and Toshiya Yamaji. Electricity load forecasting using clustering and arima model for energy management in buildings. *Japan Architectural Review*, 3(1):62–76, 2020.

[154] Shuteng Niu, Yongxin Liu, Jian Wang, and Houbing Song. A decade survey of transfer learning (2010–2020). *IEEE Transactions on Artificial Intelligence*, 1(2):151–166, 2020.

[155] Albert B Novikoff. On convergence proofs for perceptrons. Technical report, STANFORD RESEARCH INST MENLO PARK CA, 1963.

[156] European Society of Radiology (ESR) eu-affairs@ myesr. org. The new eu general data protection regulation: what the radiologist should know. *Insights into Imaging*, 8:295–299, 2017.

[157] Eva Ostertagová and Oskar Ostertag. The simple exponential smoothing model. In *The 4th International Conference on Modelling of Mechanical and Mechatronic Systems, Technical University of Košice, Slovak Republic, Proceedings of conference*, pages 380–384, 2011.

[158] Akito Ozawa, Ryota Furusato, and Yoshikuni Yoshida. Determining the relationship between a household's lifestyle and its electricity consumption in japan by analyzing measured electric load profiles. *Energy and Buildings*, 119: 200–210, 2016.

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

[160] John Paparrizos and Luis Gravano. k-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 1855–1870, 2015.

[161] Karl Pearson. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London,*

Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 50 (302):157–175, 1900.

[162] Paweł Pełka and Grzegorz Dudek. Pattern-based long short-term memory for mid-term electrical load forecasting. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[163] Daniel Perdices, Jorge E López de Vergara, and Javier Ramos. Deep-fda: Using functional data analysis and neural networks to characterize network services time series. *IEEE Transactions on Network and Service Management*, 18(1):986–999, 2021.

[164] Luis Pérez-Lombard, José Ortiz, and Christine Pout. A review on buildings energy consumption information. *Energy and buildings*, 40(3):394–398, 2008.

[165] Robi Polikar. Ensemble learning. In *Ensemble machine learning*, pages 1–34. Springer, 2012.

[166] Henrique Pötter, Stephen Lee, and Daniel Mossé. Towards privacy-preserving framework for non-intrusive load monitoring. In *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*, pages 259–263, 2021.

[167] Daniel Precioso and David Gómez-Ullate. Nilm as a regression versus classification problem: the importance of thresholding. *arXiv preprint arXiv:2010.16050*, 2020.

[168] F. Rahimi and A. Ipakchi. Demand response as a market resource under the smart grid paradigm. *IEEE Transactions on Smart Grid*, 1:82–88, 2010.

[169] Amin Rajabi, Mohsen Eskandari, Mojtaba Jabbari Ghadi, Li Li, Jiangfeng Zhang, and Pierluigi Siano. A comparative study of clustering techniques for electrical load pattern segmentation. *Renewable and Sustainable Energy Reviews*, 120:109628, 2020.

[170] James O Ramsay and CJ1125714 Dalzell. Some tools for functional data analysis. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(3):539–561, 1991.

[171] JO Ramsay and BW Silverman. Tools for exploring functional data. *Functional data analysis*, pages 19–35, 2005.

[172] Sebastian Raschka. *Python machine learning*. Packt publishing ltd, 2015.

[173] Chotirat Ratanamahatana and E. Keogh. Everything you know about dynamic time warping is wrong. 01 2004.

[174] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[175] Kunal Roy, Supratik Kar, and Rudra Narayan Das. *Understanding the Basics of QSAR for Applications in Pharmaceutical Sciences and Risk Assessment*. Academic Press, 2015.

[176] Patrick Royston. Multiple imputation of missing values. *The Stata Journal*, 4(3):227–241, 2004.

[177] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 41(4):801–814, 2018.

[178] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning Internal Representations by Error Propagation*. MIT Press, 1988.

[179] David Ruppert. The elements of statistical learning: Data mining, inference, and prediction. *Journal of the American Statistical Association*, 99(466):567–567, 2004.

[180] Emily M Ryan and Thomas F Sanquist. Validation of building energy modeling tools under idealized and realistic conditions. *Energy and buildings*, 47: 375–382, 2012.

[181] Seunghyoung Ryu, Jaekoo Noh, and Hongseok Kim. Deep neural network based demand side short term load forecasting. *Energies*, 10(1):3, 2016.

[182] Julio R. Gómez Sarduy, Katia Gregio Di Santo, and Marco Antonio Saidel. Linear and non-linear methods for prediction of peak load at university of são paulo. *Measurement*, 78:187–201, 2016.

[183] Alexis Sardá-Espinosa. Time-series clustering in r using the dtwclust package. *The R Journal*, 11:22, 06 2019. doi: 10.32614/RJ-2019-023.

[184] Aven Satre-Meloy, Marina Diakonova, and Philipp Grünewald. Cluster analysis and prediction of residential peak demand profiles using occupant activity data. *Applied Energy*, 260:114246, 2020.

[185] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 31(9):3400–3413, 2019.

[186] Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, pages 461–464, 1978.

[187] Ismail Shah and Francesco Lisi. Day-ahead electricity demand forecasting with nonparametric functional models. In *2015 12th International Conference on the European Energy Market (EEM)*, pages 1–5. IEEE, 2015.

[188] Ismail Shah and Francesco Lisi. Forecasting of electricity price through a functional prediction of sale and purchase curves. *Journal of Forecasting*, 39 (2):242–259, 2020.

[189] Joan G Staniswalis and J Jack Lee. Nonparametric regression analysis of longitudinal data. *Journal of the American Statistical Association*, 93(444): 1403–1418, 1998.

[190] Guanglu Sun, Lili Liang, Teng Chen, Feng Xiao, and Fei Lang. Network traffic classification based on transfer learning. *Computers & electrical engineering*, 69:920–927, 2018.

[191] MA Syakur, BK Khotimah, EMS Rochman, and Budi Dwi Satoto. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. In *IOP conference series: materials science and engineering*, volume 336, page 012017. IOP Publishing, 2018.

[192] Afaf Taïk and Soumaya Cherkaoui. Electrical load forecasting using edge computing and federated learning. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2020.

[193] Yansong Tang, Yi Wei, Xumin Yu, Jiwen Lu, and Jie Zhou. Graph interaction networks for relation transfer in human activity videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(9):2872–2886, 2020.

[194] Oscar Trull, J Carlos García-Díaz, and Alicia Troncoso. One-day-ahead electricity demand forecasting in holidays using discrete-interval moving seasonalities. *Energy*, 231:120966, 2021.

[195] Nguyen Truong, Kai Sun, Siyao Wang, Florian Guitton, and YiKe Guo. Privacy preservation in federated learning: An insightful survey from the gdpr perspective. *Computers & Security*, 110:102402, 2021.

[196] Israr Ullah, Rashid Ahmad, and DoHyeun Kim. A prediction mechanism of energy consumption in residential buildings using hidden markov model. *Energies*, 11(2):358, 2018.

[197] Javier Valdes, Yunesky Masip Macia, Wolfgang Dorner, and Luis Ramirez Camargo. Unsupervised grouping of industrial electricity demand profiles: Synthetic profiles for demand-side management applications. *Energy*, 215: 118962, 2021.

[198] Veeramanikandan, Suresh Sankaranarayanan, Joel J.P.C. Rodrigues, Vijayan Sugumaran, and Sergei Kozlov. Data flow and distributed deep neural network based low latency iot-edge computation model for big data environment. *Engineering Applications of Artificial Intelligence*, 94:103785, 2020.

[199] J Vermaak and EC Botha. Recurrent neural networks for short-term load forecasting. *IEEE Transactions on Power Systems*, 13(1):126–132, 1998.

[200] Joaquim L Viegas, Paulo R Esteves, and Susana M Vieira. Clustering-based novelty detection for identification of non-technical losses. *International Journal of Electrical Power & Energy Systems*, 101:301–310, 2018.

[201] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 1698–1707. IEEE, 2020.

[202] 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.

[203] Lan Wang, Eric WM Lee, Syed Asad Hussian, Anthony Chun Yin Yuen, and Wei Feng. Quantitative impact analysis of driving factors on annual residential building energy end-use combining machine learning and stochastic methods. *Applied Energy*, 299:117303, 2021.

[204] Yi Wang, Qixin Chen, Tao Hong, and Chongqing Kang. Review of smart meter data analytics: Applications, methodologies, and challenges. *IEEE Transactions on Smart Grid*, 10(3):3125–3148, 2018.

[205] Zeyu Wang and Ravi S Srinivasan. A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models. *Renewable and Sustainable Energy Reviews*, 75: 796–808, 2017.

[206] Zeyu Wang, Yueren Wang, Ruochen Zeng, Ravi S. Srinivasan, and Sherry Ahrentzen. Random forest based hourly building energy prediction. *Energy and Buildings*, 171:11–25, 2018.

[207] Zheng Wang, Yangqiu Song, and Changshui Zhang. Transferred dimensionality reduction. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 550–565. Springer, 2008.

[208] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020.

[209] Rui Xia, Xuelei Hu, Jianfeng Lu, Jian Yang, and Chengqing Zong. Instance selection and instance weighting for cross-domain sentiment classification via pu learning. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 2176–2182, 2013.

[210] Feng Xu, Jianfei Yu, and Rui Xia. Instance-based domain adaptation via multiclustering logistic approximation. *IEEE Intelligent Systems*, 33(1):78–88, 2018.

[211] Jieyan Xu, Xuyuan Kang, Zheng Chen, Da Yan, Siyue Guo, Yuan Jin, Tianyi Hao, and Rongda Jia. Clustering-based probability distribution model for monthly residential building electricity consumption analysis. In *Building Simulation*, volume 14, pages 149–164. Springer, 2021.

[212] Robert Alan Yaffee and Monnie McGee. *An introduction to time series analysis and forecasting: with applications of SAS® and SPSS®*. Elsevier, 2000.

[213] Melek Yalcintas and U Aytun Ozturk. An energy benchmarking model based on artificial neural network method utilizing us commercial buildings energy consumption survey (cbecs) database. *International Journal of Energy Research*, 31(4):412–421, 2007.

[214] Ke Yan, Xinlu Guo, Zhiwei Ji, and Xiaokang Zhou. Deep transfer learning for cross-species plant disease diagnosis adapting mixed subdomains. *IEEE/ACM transactions on computational biology and bioinformatics*, 2021.

[215] Ailing Yang, Weide Li, and Xuan Yang. Short-term electricity load forecasting based on feature selection and least squares support vector machines. *Knowledge-Based Systems*, 163:159–173, 2019.

[216] Hongwei Yang, Hui He, Weizhe Zhang, and Xiaochun Cao. Fedsteg: A federated transfer learning framework for secure image steganalysis. *IEEE Transactions on Network Science and Engineering*, 2020.

[217] 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.

[218] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. Vertical federated learning. In *Federated Learning*, pages 69–81. Springer, 2020.

[219] Shengwen Yang, Bing Ren, Xuhui Zhou, and Liping Liu. Parallel distributed logistic regression for vertical federated learning without third-party coordinator. *arXiv preprint arXiv:1911.09824*, 2019.

[220] Youlong Yang, Jinxing Che, Chengzhi Deng, and Li Li. Sequential grid approach based support vector regression for short-term electric load forecasting. *Applied energy*, 238:1010–1021, 2019.

[221] Yun Yang, Xinfa Li, Pei Wang, Yuelong Xia, and Qiongwei Ye. Multi-source transfer learning via ensemble approach for initial diagnosis of alzheimer's disease. *IEEE Journal of Translational Engineering in Health and Medicine*, 8:1–10, 2020.

[222] Junwen Yao, Jonas Mueller, and Jane-Ling Wang. Deep learning for functional data analysis with adaptive basis layers. In *International Conference on Machine Learning*, pages 11898–11908. PMLR, 2021.

[223] Yulong Ye, Qiuzhen Lin, Lijia Ma, Ka-Chun Wong, Maoguo Gong, and Carlos A Coello Coello. Multiple source transfer learning for dynamic multiobjective optimization. *Information Sciences*, 2022.

[224] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pages 268–282. IEEE, 2018.

[225] Baran Yildiz, Jose I Bilbao, Jonathon Dore, and Alistair B Sproul. Recent advances in the analysis of residential electricity consumption and applications of smart meter data. *Applied Energy*, 208:402–427, 2017.

[226] Baran Yildiz, Jose I Bilbao, and Alistair B Sproul. A review and analysis of regression and machine learning models on commercial building electricity load forecasting. *Renewable and Sustainable Energy Reviews*, 73:1104–1122, 2017.

[227] Selin Yilmaz, Jonathan Chambers, and Martin Kumar Patel. Comparison of clustering approaches for domestic electricity load profile characterisation-implications for demand side management. *Energy*, 180:665–677, 2019.

[228] Moustafa Youssef, Matthew Mah, and Ashok Agrawala. Challenges: device-free passive localization for wireless environments. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 222–229, 2007.

[229] Jungwon Yu, Hansoo Lee, Yeongsang Jeong, and Sungshin Kim. Linguistic fuzzy modeling approach for daily peak load forecasting. In *2013 International Conference on Fuzzy Theory and Its Applications (iFUZZY)*, pages 116–121. IEEE, 2013.

[230] Lean Yu, Yaqing Zhao, Ling Tang, and Zebin Yang. Online big data-driven oil consumption forecasting with google trends. *International Journal of Forecasting*, 35(1):213–223, 2019.

[231] Haixiang Zang, Ruiqi Xu, Lilin Cheng, Tao Ding, Ling Liu, Zhinong Wei, and Guoqiang Sun. Residential load forecasting based on lstm fusing self-attention mechanism with pooling. *Energy*, 229:120682, 2021.

[232] P. Zeng and M. Jin. Peak load forecasting based on multi-source data and day-to-day topological network. *IET Generation, Transmission Distribution*, 12:1374–1381, 2018.

[233] Bing Zhang, Jhen-Long Wu, and Pei-Chann Chang. A multiple time series-based recurrent neural network for short-term load forecasting. *Soft Computing*, 22(12):4099–4112, 2018.

[234] 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.

[235] Ning Zhang, Zhiying Li, Xun Zou, and Steven M Quiring. Comparison of three short-term load forecast models in southern california. *Energy*, 189: 116358, 2019.

[236] 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.

[237] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[238] Peilin Zhao, Jialei Wang, Pengcheng Wu, Rong Jin, and Steven CH Hoi. Fast bounded online gradient descent algorithms for scalable kernel-based online learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1075–1082, 2012.

[239] Tongya Zheng, Gang Chen, Xinyu Wang, Chun Chen, Xingen Wang, and Sihui Luo. Real-time intelligent big data processing: technology, platform, and applications. *Science China Information Sciences*, 62(8):1–12, 2019.

[240] Mingjun Zhong, Nigel Goddard, and Charles Sutton. Latent bayesian melding for integrating individual and population models. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pages 3618–3626, 2015.

[241] Joey Tianyi Zhou, Sinno Jialin Pan, and Ivor W Tsang. A deep learning framework for hybrid heterogeneous transfer learning. *Artificial Intelligence*, 275:310–328, 2019.

[242] Jun Zhou, Xiaolong Li, Peilin Zhao, Chaochao Chen, Longfei Li, Xinxing Yang, Qing Cui, Jin Yu, Xu Chen, Yi Ding, et al. Kunpeng: Parameter server based distributed learning systems and its applications in alibaba and ant financial. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1693–1702, 2017.

[243] Shan Zhou and Marilyn A Brown. Smart meter deployment in europe: A comparative case study on the impacts of national policy schemes. *Journal of Cleaner Production*, 144:22–32, 2017.

[244] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 928–936, 2003.