Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria dell'Informazione
Curriculum in Biomedical, Electronics and Telecommunications Engineering

Deep Learning Techniques for Edge-Centric Non-Intrusive Load Monitoring

Ph.D. Dissertation of:
Giulia Tanoni

Advisor:
Prof. Emanuele Principi

Curriculum Supervisor:
Prof. Franco Chiaraluce

XXXVI Cycle

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria dell'Informazione
Curriculum in Biomedical, Electronics and Telecommunications Engineering

Deep Learning Techniques for Edge-Centric Non-Intrusive Load Monitoring

Ph.D. Dissertation of:
Giulia Tanoni

Advisor:
Prof. Emanuele Principi

Curriculum Supervisor:
Prof. Franco Chiaraluce

XXXVI Cycle

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
dell'Informazione
Facoltà di Ingegneria
Via Brecce Bianche – 60131 Ancona (AN), Italy

a Hollie, Jim e Jon!

Acknowledgments

Come discutevamo con un mio caro amico in un’aula dell’Università di Strathclyde, il percorso di dottorato può essere rappresentato graficamente con una sinusoide. Si inizia e ci si ritrova subito sulla cresta dell’onda, pieni di aspettative, ci si immagina di inventare il metodo che rivoluzionerà il mondo il giorno successivo. Poi un codice non funziona, i risultati non tornano e peggio ancora la tua idea che credevi super innovativa è già stata sviluppata da altri mille ricercatori sparsi per il mondo. Da qui, la discesa verso il minimo con inadeguatezza, senso di fallimento, e reali considerazioni sul cambiare mestiere. Come si risale alla cresta successiva? Innanzitutto con la determinazione, che non manca mai e la voglia di portare fino in fondo un’idea in cui si crede. Per questo ringrazio me stessa. Si risale con l’appoggio di una persona di fiducia, con le sue critiche costruttive con cui dimostra di credere in te e ti stimola a migliorarti continuamente. Salire degli scalini è più facile che affrontare una salita ripida. Degli scalini che mi hanno dato sicurezza e su cui fermamente ho potuto mettere il piede per fare il passo successivo. Per questo ringrazio tanto il mio tutor. Ringrazio i miei genitori, mio fratello e le mie amiche per aver camminato accanto a me, ascoltandomi e interessandosi al mio lavoro e alle mie passioni. Grazie per le parole di conforto che hanno reso meno rarefatta l’aria verso la cima. Ringrazio la mia cara compagna musica, sempre, e le persone con cui la condivido. Anche quelle che non ci sono più ma la cui musica suona ancora. Ringrazio Stefano, per avermi offerto la possibilità di lavorare nella ricerca e sognare qualcosa di migliore, che più si va avanti e più è difficile farlo. Ringrazio i miei supervisori a Strathclyde per avermi accolto in una realtà che mi ha aperto gli occhi per guardare meglio ciò che mi piaceva. Ringrazio i colleghi ma più che altro amici di Glasgow. Uno in particolare, che è anche il mio English coach. In ultimo, ma non assolutamente per importanza, ringrazio Enrico per avermi tenuto stretta la mano in quest’ultimo anno che ha richiesto lo slancio più grande. Lo ringrazio per aver rispettato le mie emozioni, ed averle comprese con la sua immensa sensibilità. Lo ringrazio perché mentre camminiamo verso la fine vedo una casetta gialla e sorrido. Condivido così tanto con lui che è come se ci fosse sempre stato.

Ancona, Febbraio 2024

Giulia Tanoni

Abstract

Climate change is one of the most significant challenges faced in this century. To limit the rise in global average temperatures below 1.5 °C, it is crucial to decrease the electrical energy usage. Therefore, it is vital to promote energy efficiency through sustainable practices. Devices should be used efficiently to avoid energy waste. In the residential setting, individuals can significantly contribute to energy saving, especially if they are aware of their consumption. The constant availability of energy consumption profiles has led to the development of advanced techniques to monitor loads inside buildings and provide residential users with improved awareness of their energy consumption and usage habits. One such technique is Non-Intrusive Load Monitoring (NILM), which detects the states of appliances and estimates the power consumption of individual loads in the building based only on the building’s aggregate meter readings. Nowadays, most of the approaches proposed in the literature are based on deep learning, which has proven superior to other methods. Nonetheless, they still have to deal with aspects related to real-world applicability. Firstly, there is the issue of the availability of labeled datasets. Labels should be provided by annotators, who are often end-user, but this process is time-consuming and prone to errors. Second, computation is usually done in the cloud, which is far from where the data are acquired. This requires data transmission and can result in latency in the service output. To mitigate the above issues, this thesis proposes several methodologies that follow the Human-Centred Computing and the Edge Computing paradigms. As a consequence, the developed strategies aim to lighten the effort requested to the user for providing labels while enhancing the performance. At the same time, the computation is lightened by reducing algorithms complexity while maintaining performance. The methods have been developed and evaluated on publicly available datasets, demonstrating their superiority compared to benchmark strategies. Moreover, the final performance is increased, even with less data and simpler structures. Future directions considers to train networks locally to promote adaptability and reliability. Additionally, hybrid monitoring strategies can be investigated and integrated with energy management systems or demand-response programs based on the user requirements.

Abstract

Il cambiamento climatico è una delle sfide più significative affrontate in questo secolo. Per limitare l’aumento della temperatura media globale al di sotto di 1.5 °C, è fondamentale ridurre l’uso di energia elettrica ed è vitale promuovere l’efficienza energetica. I dispositivi elettronici dovrebbero essere utilizzati in modo efficiente per evitare sprechi di energia. Nell’ambito residenziale, gli utenti possono contribuire significativamente al risparmio energetico, soprattutto se consapevoli dei loro consumi. La costante disponibilità di dati di consumo energetico ha portato allo sviluppo di tecniche avanzate per monitorare i carichi all’interno degli edifici e fornire agli utenti residenziali una maggiore consapevolezza delle loro abitudini di consumo elettrico. Il monitoraggio non intrusivo del carico stima lo stato e il consumo energetico dei singoli elettrodomestici nell’edificio, basandosi solo sulle letture aggregate del contatore. Oggigiorno, la maggior parte degli approcci proposti in letteratura si basa sul Deep Learning, poiché si è dimostrato superiore ad altri metodi inizialmente sviluppati. Tuttavia, devono essere ancora affrontati aspetti legati all’applicabilità nel mondo reale. In primo luogo, c’è il problema della disponibilità di dati annotati per addestrare approcci supervisionati. Chi fornisce le annotazioni spesso coincide con l’utente finale e il processo di annotazione può essere lungo, scomodo e soggetto ad errori. In secondo luogo, l’inferenza viene solitamente eseguita nel cloud su macchine ad alte risorse, quindi lontano da dove vengono acquisiti i segnali. Questo richiede la trasmissione dei dati e può comportare ritardi del servizio e problemi di privacy. Per mitigare le suddette problematiche, questa tesi propone metodologie basate sui paradigmi dello Human-centred Computing e dell’ Edge Computing. Le strategie sviluppate mirano a ridurre lo sforzo richiesto all’utente per fornire le annotazioni mantenendo stabili o migliorando le prestazioni. Inoltre, i metodi mirano a diminuire la complessità strutturale e computazionale delle architetture utilizzate. Gli approcci sono stati sviluppati e valutati su dataset pubblici, dimostrando la loro superiorità rispetto allo stato dell’arte. Le prestazioni finali risultano superiori utilizzando meno dati, annotazioni più deboli e strutture di rete più semplici che risultano adatte per l’installazione su dispositivi a basse risorse. Ricerche future considerano di addestrare le reti neurali localmente, per promuovere l’adattabilità e l’affidabilità del monitoraggio. Inoltre, possono essere sviluppate strategie di monitoraggio ibride e integrate con sistemi di gestione dell’energia.

Contents

I. Preliminaries **1**

1. Introduction **3**

2. Background and Contributions **9**
2.1. Problem Statement 9
2.2. General NILM framework 12
2.3. Datasets 15
2.4. Related Works 17
2.4.1. Power Profile Reconstruction 17
2.4.2. Appliance State Classification 18
2.4.3. Multi-task approaches 19
2.5. Open Issues and Contributions 19

3. Edge-Centric Non-Intrusive Load Monitoring Framework **21**
3.1. Convolutional Recurrent Neural Network 21
3.2. Weak Supervision 24
3.3. Transfer Learning 26
3.4. Knowledge Distillation 27
3.5. Continual Learning 28

II. User-centred NILM Methods **31**

4. Introduction **35**

5. Multi-Label Appliance Classification with Weakly Labeled Data **39**
5.1. Proposed methodology 40
5.1.1. Neural Network Architecture 42
5.1.2. Learning 43
5.1.3. Post-Processing 44
5.2. Experimental setup 45
5.2.1. Dataset 45
5.2.2. Benchmark Methods 46
5.2.3. Evaluation Metrics 46

5.2.4. Experimental procedure 47
5.2.5. Post-processing 49
5.2.6. Complexity Details 50
5.3. Results experiment 1: Fixed amount of weakly labeled data . . 51
5.3.1. UK-DALE 51
5.3.2. REFIT 54
5.4. Results experiment 2: Fixed amount of strongly labeled data . 56
5.4.1. UK-DALE 57
5.4.2. REFIT 58
5.5. Results experiment 3: Mixed training set 59

6. A Multiple Instance Regression Approach to Electrical Load Disaggregation 61
6.1. Proposed Methodology 61
6.2. Neural Network and Learning 62
6.3. Experiments 63
6.3.1. Experimental procedure 64
6.3.2. Hyperparameters 64
6.3.3. Evaluation metrics 65
6.4. Results 65

7. Weakly Supervised Transfer Learning for Multi-label Appliance Classification 69
7.1. Proposed Methodology 70
7.2. Experimental Setting 72
7.2.1. Dataset 72
7.2.2. Experimental setup 72
7.3. Results and Discussion 74

8. A Weakly Supervised Active Learning Framework for Non-Intrusive Load Monitoring 77
8.1. Learning Strategy 78
8.2. Weakly Supervised AL Framework 78
8.3. Acquisition function 81
8.4. Experimental Setting 83
8.4.1. Dataset 83
8.4.2. Experiments setup 83
8.4.3. Benchmark method 84
8.4.4. Evaluation metrics 85
8.5. Results 85
8.5.1. Semi-supervised benchmark results 85
8.5.2. Weakly Supervised AL Performance 87

8.5.3. Complexity Details 94

9. Discussion **95**

III. Low-Complexity NILM Methods **97**

10. Introduction **101**

11. Knowledge Distillation for Scalable Non-Intrusive Load Monitoring **105**
11.1. Proposed Methodology 105
11.1.1. Teacher Learning 107
11.1.2. Student Knowledge Distillation 108
11.1.3. Neural Network Architectures 109
11.2. Experimental Setup 110
11.2.1. Dataset 110
11.2.2. Hyperparameters 111
11.2.3. Architecture Complexity Evaluation 111
11.2.4. Benchmark Methods 112
11.2.5. Evaluation Metrics 113
11.3. Results and Discussion 113
11.3.1. Window Length Impact on Teacher Performance 114
11.3.2. Student Distillation Results 118
11.3.3. Comparison with Benchmark Methods 121

12. Improving Knowledge Distillation through Explainability Guided Learning **125**
12.1. Proposed Methodology 126
12.2. Knowledge Distillation 126
12.3. Feature Importance Map Generation 127
12.4. Explainability Guided Learning 128
12.5. Experimental setup 130
12.5.1. Datasets 130
12.5.2. Training Procedure 130
12.6. Results 131

13. Appliance incremental learning for Non-Intrusive Load Monitoring **135**
13.1. Proposed Methodology 136
13.1.1. Neural Network Architecture 137
13.2. Appliance-Incremental Learning Framework 137
13.2.1. Baseline learning 137
13.2.2. Adaptation learning 137
13.2.3. Dynamic Layer Selection 138

13.3. Experiments 140
13.3.1. Dataset 140
13.3.2. Evaluation metrics 140
13.3.3. Hyperparameters 140
13.3.4. Experimental procedure 141
13.4. Results 142

14. Discussion **145**

IV. Conclusions **147**

15. Conclusions and future works **149**
15.1. Future perspectives 150

V. Appendix **151**
1. Data Preparation 153
2. Benchmark Approaches 154
2.1. Long-Short Term Memory Network 154
2.2. Temporal Convolutional Network 154
2.3. Sequence-to-point 155

List of Figures

1.1. Electricity consumption by sector expressed in Exajoule. A total of 4 Exajoule and 82 Exajoule based on [1], was in 1973 and 2019 respectively. Other categories includes agriculture and fishing. . 4
1.2. Load Monitoring approaches. 5

2.1. Example of multi-label appliance classification task. For one window of aggregate power signal the aim is to localize in time for more than one appliance, where and which appliance is active. 10
2.2. Example of appliance power profile reconstruction task. For one window of aggregate power signal the aim is to estimate the power profiles of each appliance of interest. 10
2.3. Examples of output signals for regression (second row) and classification (third raw), for the same input window (first row). . . 12
2.4. NILM framework 12
2.5. Different input processing approaches. The signals are respectively the aggregate (orange) and appliance-level consumption (blue). The bar evidences the different lengths of the output sequence for the same input. 14
2.6. Example of a daily consumption of House 1 from UK-DALE. . 15
2.7. Example of daily consumption from AMPds. 17

3.1. Overall approach adopted for Edge-Centric Non-Intrusive Load Monitoring. 22
3.2. General CRNN architecture. For the sake of conciseness, only one convolutional block is graphically reported. The example regards the multi-label classification task, thus the sigmoid function is adopted in the fully connected layer to produce the outputs. 23
3.3. Weak Supervision. A binary classification example is represented. 25
3.4. Schematic representation for transfer learning. The example refers to the case when the task to perform is the same but data belong to different domains. All the blocks that learnt common low-level features about the task are transferred to the new network and only the last layer is fine-tuned on the new domain data. 26

List of Figures

3.5. Knowledge Distillation based on Teacher-Student strategy for a model compression application. The $\mathcal{L}$ function is a generic loss function. 27

5.1. Schematic representation NILM formulated as Multiple-Instance Learning. KE: Kettle. MW: Microwave. DW: Dishwasher. . . . 40

5.2. An example of aggregate segment from house 2 of REFIT with the related labels. The weak label is represented by the presence of the tag with the appliance name, meaning that inside the window the appliance is active at least one time. The dimension of the coloured segment defines the ON- and OFF-time of the appliance activation. KE, MW, FR, WM, and DW stand respectively for Kettle, Microwave, Fridge, Washing Machine, Dishwasher. 41

5.3. Block scheme of the proposed approach. The network takes as input the aggregate power related to the j-th window and produces two outputs, one from the instance layer ($\hat{\mathbf{S}}_j$) and one from the bag layer ($\hat{\mathbf{w}}_j$). The multiplication is related to clip smoothing. T is length of input and output window, FCL stands for Fully Connected Layer, K is number of appliances. 43

5.4. Difference between F_1-scores of each appliance, F_1-micro, and TECA of the proposed method and S-CRNN for UK-DALE for the different percentages of strongly labeled data (Section 5.3). 51

5.5. Training loss and validation loss and F_1-score for the experiment related to 40% strong data and 100% weak data for UK-DALE. Vertical bar indicates the early stopping epoch. 53

5.6. Difference between F_1-scores of each appliance, F_1-micro, and TECA of the proposed method and S-CRNN for REFIT for the different percentages of strongly labeled data (Section 5.3). . . 56

6.1. The proposed architecture for MIR-based NILM. 62

6.2. Ground-truth and estimated active power for the proposed and comparative methods for different percentages of strongly labeled data (shown in brackets). 68

7.1. Transfer learning with weak supervision. The model is trained with *source* domain data. Then the CNN blocks are frozen, and the remaining layers are fine-tuned with *target* domain data. . . 71

7.2. Classification predictions produced by each pre-trained model after fine-tuning with weakly labeled data. Data is from REFIT house 2. AGG: Aggregate. 75

8.1. Weakly Supervised AL Scheme. Each block corresponds to an element of the framework. The Convolutional Recurrent Neural Network (CRNN) model generates both strong and weak predictions. During the AL process, strong predictions are used to evaluate the current model, while weak predictions serve as input for the acquisition function. The acquisition function selects the windows to be labelled based on the uncertainty of the network predictions. The most uncertain windows are chosen, suggested to the user for annotation, and then incorporated into the fine-tuning set for the subsequent fine-tuning phase. A detailed description of the entire framework can be found in Section 8.2. 79

8.2. Observed uncertainty levels in Scenario 1 (top) and Scenario 2 (bottom) for the whole query pool of House 4 bags. 92

8.3. Observed ratio of uncertainty between kettle and microwave in Scenarios 1 (top) and 2 (bottom), when using mean (left) and maximum (right) uncertainty across present appliances. 93

8.4. AL curve obtained at REFIT House 4 in Scenario 2 when averaging uncertainty across present appliances. Original curve is smoothed using Savitsky-Golay filter of length 11 and order 3. . 94

9.1. Scatter plot for number of annotated bags vs performance expressed in terms of F_1-micro. "W": Chapter 5 ([2]), "W-TL": Chapter 7 ([3]), and "W-AL": Chapter 8 ([4]). 96

11.1. Proposed Knowledge Distillation framework for NILM. "GT" stands for Ground Truth. 106

11.2. Window analysis based on the test set: F_1-scores when the Teacher is pre-trained with REFIT. 114

11.3. Window analysis based on the test set: F_1-scores when the Teacher is pre-trained with UK-DALE. 114

11.4. Complexity-Performance comparison among benchmark methods and the proposed method. For each approach the dimension of the circle is proportional to the mean F_1-score of both D_1 scenarios. FLOPs are expressed in Millions (M) and Size is expressed in megabytes (MB). 122

List of Figures

12.1. Explanations for prediction of Washing Machine on a sample from the test set in the REFIT-to-REFIT domain adaptation scenario. a) Teacher explanation b) baseline Student explanation, displaying the inconsistent transfer of explanation knowledge c) Corrected Student explanation and prediction after explainability guided learning. Strong predictions are displayed before quantization. 133

13.1. AIL method scheme for NILM. To introduce a new appliance, adaptation learning and dynamic layer distillation are applied. The arrow from the previous to the subsequent model indicates that adaptation learning is done by using the previous model as the Teacher in the distillation. In this case, V and Q are equal to 1. 136

13.2. Performance trend with varying batch sizes for the proposed AIL approach. AVG denotes the average performance. 141

List of Tables

5.1. UK-DALE dataset characteristics. Numbers are in thousands. 45
5.2. REFIT dataset characteristics. Numbers are in thousands. . . . 45
5.3. Training hyperparameters not subject to tuning. 48
5.4. Hyperband parameters. "Max epochs" refers to epochs for the Hyperband algorithm thus the number differs from the epochs of the learning process. U is the number of GRUs, H is the number of convolutional blocks, K_e is the kernel dimension and p is the dropout probability. 49
5.5. Hyperparameters determined after tuning. 50
5.6. Maximum model size, training and test time of all the evaluated methods. 50
5.7. Results obtained on the UK-DALE and REFIT datasets by using weakly labeled data only, in terms of F_1-score (Section 5.3). . 51
5.8. Results obtained on the UK-DALE dataset related to Experiment 1. Best scores for each strong percentage are highlighted in bold. Best score among all the percentage are underlined (Section 5.3). 52
5.9. Results obtained on the REFIT dataset related to Experiment 1. Best scores for each strong percentage are highlighted in bold. Best score among all the percentage are underlined (Section 5.3). 55
5.10. Results obtained on the UK-DALE dataset related to Experiment 2. The best results obtained using the least amount of weakly labeled data are highlighted in bold. (Section 5.4) . . . 57
5.11. Results obtained on the REFIT dataset related to Experiment 2. The best results obtained using the least amount of weakly labeled data are highlighted in bold (Section 5.4). 58
5.12. Results obtained on the UK-DALE test set with mixed training set. Best scores are reported in bold (Section 5.5). 59
5.13. Results obtained on REFIT test set with mixed training set. Best scores are reported in bold (Section 5.5). 59

6.1. Results obtained for the different training conditions and addressed methods (Section 6.4). Best results for each appliance and percentage of strong labels are reported in bold. 67

List of Tables

7.1. Train, Validation and Test sets characteristics for REFIT. Number of labels is reported in thousands. SL: Strong Labels. WL: Weak Labels. 72

7.2. CRNN hyperparameters after tuning. 73

7.3. Results related to the Baseline and all the pre-training scenarios. Best results are reported in bold. Kettle: KE, Microwave: MW, Fridge: FR, Washing Machine: WM, Dishwasher: DW. 74

8.1. Uncertainty-based acquisition function example: Uncertainty levels for each appliance are calculated as per (8.3), and, maximum or mean uncertainty values are calculated based on (8.4) and (8.5), respectively. In this example, a batch of $P = 4$ most uncertain bags is chosen. 82

8.2. Fine-Tuning and Test sets characteristics for REFIT. Number of labels is reported in thousands. WL: Weak Labels. 83

8.3. Benchmark - semi supervised method [5]. Model is pre-trained using strong labels, but fine-tuned using only unlabelled data from target environment. Results of the proposed approach are shown in the following format: metric (% of activation samples added to fine-tuning dataset). 86

8.4. Results - pre-training Scenario 1. Results of the proposed approach are shown in the following format: metric (% of activation samples added to fine-tuning dataset). 88

8.5. Results - pre-training Scenario 2. Results of the proposed approach are shown in the following format: metric (% of activation samples added to fine-tuning dataset). 89

11.1. Pre-training sets characteristics. Mean Power (expressed in Watt) refers to the mean power estimated in a complete activation while the Duration (expressed in minutes) refers to the length. 110

11.2. Total number of parameters for convolutional and recurrent subparts are reported for the teacher network and each student architecture. 112

11.3. Number of FLOPs and sizes of teacher and student networks for different window lengths (in samples) and number of classes $K = 3$ and $K = 6$. 112

11.4. Performance comparison between the Teacher (D_1=REFIT) and the Student networks. Improved performance are in bold and underlined while equal performance are in bold for the reduced Student architectures. 116

11.5. Performance comparison between the Teacher (D_1=UK-DALE) and the Student networks. Improved performance are in bold and underlined while equal performance are in bold for the reduced Student architectures. 117
11.6. Performance comparison between the Teacher (D_1=REFIT) and the reduced Student networks in terms of TECA. Improved and equal performance are reported in bold for each Student architecture. 118
11.7. Performance comparison between the Teacher (D_1=UK-DALE) and the reduced Student networks in terms of TECA. Improved and equal performance are reported in bold for each Student architecture. 118
11.8. Results in terms of F_1-score of the proposed approach and benchmark methods trained with D_1=REFIT and tested on REFIT. Best results are reported in bold. 120
11.9. Results in terms of F_1-score of the proposed approach and benchmark methods trained with D_1=UK-DALE and tested on REFIT. Best results are reported in bold. 121
11.10Model Size (MB) and FLOPS (M) for the benchmark methods and the proposed approach. The model size and the number of FLOPs are calculated on all the networks used to classify $K = 6$ appliances. 121

12.1. Architecture of Teacher and Student models. 127
12.2. Training hyperparameters used for training of Student models for each of the two domain adaptation scenarios. 129
12.3. Results for the UK-DALE-to-REFIT domain adaptation scenario. 132
12.4. Results for the REFIT-to-REFIT domain adaptation scenario. 132

13.1. Re-training data characteristics in terms of active samples. . . 140
13.2. Results related to House 2. Best F_1-score when considering the same appliances are highlighted in bold. 143
13.3. Results related to House 4. Best F_1-score when considering the same appliances are highlighted in bold. 144

Part I.

Preliminaries

Chapter 1.

Introduction

Climate change is one of the most significant challenges faced in this century. To limit the rise in global average temperatures to below 1.5 °C, it is crucial to decrease the electrical energy usage. The growth in electricity consumption is primarily driven by the household and services sectors, while the industrial sector's electricity usage fluctuates with the economic cycle [6].

According to the "Key World Energy Statistics 2021" by the International Energy Agency [1], the residential sector accounts for 26.6% of energy consumption, commercial and public services account for 21.2%, and the industry has the highest consumption at 41.9%. As shown in Figure 1.1, the residential and commercial sectors had an increase between 1973 and 2019. Residential consumption in particular has been on the rise due to population growth and the increasing use of technology in daily life.

Renewable energy sources are growing as well. Renewable electricity capacity additions reached an estimated 507 GW in 2023, almost 50% higher than in 2022. Thus, 2023 marks a step change for renewable power growth over the next five years[7]. Anyway, in 2021 it was estimated to be only the 10.8% of the total world energy production together with wind and geothermal sources [1, 8], thus, they are still insufficient to meet the energy demand. Therefore, it is vital to promote energy efficiency through sustainable practices. This can be achieved by efficiently using devices and avoiding energy waste. In the residential setting, individuals can significantly contribute to energy saving, especially if they are aware of their consumption. Research has shown that users who are conscious of their energy consumption are more likely to adopt efficient technologies and save energy for both financial and environmental reasons [9]. Moreover, active user participation can potentially enhance a household's energy flexibility, leading to energy savings of up to 30% [10]. Evidence suggests that energy awareness motivates end-users to buy energy-efficient products [11], which could encourage users to actively participate in energy conservation and invest in devices that yield future energy and monetary savings. A recent review emphasized that providing effective consumption feedback is another way to engage users actively in the long term [12]. The study's findings underscore

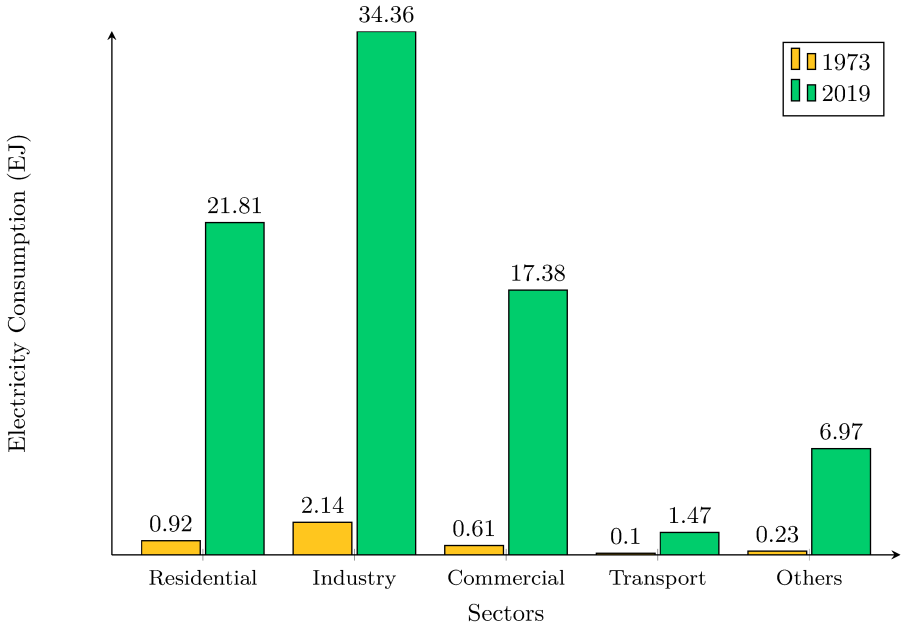
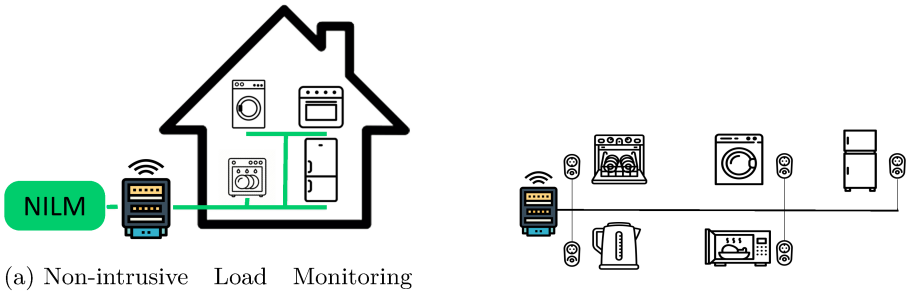


Figure 1.1.: Electricity consumption by sector expressed in Exajoule. A total of 4 Exajoule and 82 Exajoule based on [1], was in 1973 and 2019 respectively. Other categories includes agriculture and fishing.

the need to develop more user-centered strategies and technologies.

Energy awareness can be supported by energy monitoring [13], particularly through Load Monitoring, which provides detailed consumption information. In the Smart Grid ecosystem, the Advanced Metering Infrastructure facilitates communication between utilities and users through bi-directional communication [14]. By 2025, it is expected that most European countries will have implemented Smart Meter roll-out to at least 80% of consumers [15]. Smart meters enable remote measurement and management of a building's electricity consumption by interfacing with the grid [16], providing new opportunities for energy service providers to offer real-time personalized energy services to users within their homes [17]. Additionally, smart meter readings can be used to trace energy usage and propose strategies for saving energy and balancing energy supply and demand.

The constant availability of energy consumption profiles has led to the development of advanced techniques to monitor loads inside buildings and provide residential users with improved awareness of their energy consumption and usage habits. One such technique is Non-Intrusive Load Monitoring (NILM), which detects the states of loads and estimates the power consumption of individual loads in the building based only on the building's aggregate meter



(a) Non-intrusive Load Monitoring (NILM)

(b) Intrusive Load Monitoring (ILM)

Figure 1.2.: Load Monitoring approaches.

readings. Monitoring is software-based thus only one sensor is necessary, that can coincide with the smart meter already installed in the house or an additional meter. The other possibility is Intrusive Load Monitoring (ILM) that relies on the installation of as many sensors as the appliances to be monitored. Associated with this hardware-based approach, there are sensors' cost and possible difficulties or impossibilities to place the sensor on the appliance plug. In Figure 1.2a and Figure 1.2b a graphical representation describes the two different monitoring approaches. Thus, NILM has become a very active area of research with widespread smart meter installations in the residential sector. According to [18], NILM approaches that use signals with sampling frequency lower or equal to 1 Hz are considered low-frequency while for greater frequencies approaches are considered high-frequency. Since high-frequency data are related to costly sensors and they are not easily available in practical scenarios, the focus of this work is on low-frequency approaches.

After the seminal work proposed by Hart [19], several strategies have been adopted for NILM, but nowadays most of the approaches proposed in the literature are based on deep learning, demonstrating their superiority over other methodologies. Nonetheless, they still have to deal with aspects related to real-world applicability.

The literature works have mainly focused on performance optimization, taking advantage of supervised learning. They rely on the availability of large enough quantity of labeled data, that in practical scenarios are difficult to obtain. Although the main consumption is available through the smart meter, ON and OFF time or power consumption signals of each appliance are available if sensors are installed in the house or there is an annotator that provides activations timing. Another aspect, common for deep learning approaches, is the ability to generalize in unseen environment and different data domain. Specifically for NILM, moving from a data domain (represented by signals from one household) to another (represented by power signals from a different house)

can lead to a significant drop in performance. This depends on differences in users' habits, type, and number of appliances installed in the house, etc..

The end-user, who will directly use the NILM service in the residential sector, can also play the role of annotator. This is because he is the closest to the appliances and their usage patterns and frequency depend on its habits. Also, the user can participate in annotation collection to promote transfer learning strategies and alleviate undesired malfunctions of the NILM service.

In this view, the user can play a key role if included in the annotation process and until now only few works focuses marginally on its role [20, 21].

Regarding the NILM service, although it is provided at the edge of the network, still relies on a computation performed far from the user. Most of the published NILM approaches employed networks with millions of parameters that require high computational resource hardware. Due to the large diffusion of technological devices in human daily life and pervasive computing [22], computation can move closer to the users, to avoid data transmission, bandwidth, privacy, and latency problems. Only a few works in NILM focused on this aspect and without considering the practical applicability in unknown domains where these algorithms will practically perform.

For the above reasons, this work aims to develop new deep learning strategies for NILM, (i) to focus more on the figure of the user, considering its figure along the development process by simplifying its role, and (ii) to focus more on a computation localized at the edge of the network considering practical implications related to different power signal domains. These objectives are pursued by developing new NILM techniques following the Human-Centred Computing and the Edge Computing paradigms, which converge into the Edge-Centric Computing paradigm [23]. In this direction, the figure of the end-user is included longitudinally along the development process of NILM strategies, and the computational burden of the networks will be optimized to enable the computation closer to the user.

The rest of the thesis is organized as follow. Part I of the thesis continues with Chapter 2 that describes in detail the background and the contributions of the present work. Specifically, the formulation of the NILM problem, the general framework and background about datasets and works with the research gaps and how this work aims to fill them. Chapter 3 provides details of the deep learning techniques adopted in this work to develop methodologies and meet paradigms requirements.

Part II treats the methodologies proposed for user-centred NILM development. Chapter 4 collects literature on labeling effort reduction while Chapter 5, Chapter 6, Chapter 7 and Chapter 8 explain in details the methodologies proposed to fill the related research gaps. A conclusive discussion is exposed in Chapter 9.

Part III regards the methodologies for the development of low complexity NILM approaches. Chapter 10 presents an overview about recent techniques adopted for complexity reduction in NILM literature. Then, in Chapter 11, Chapter 12 and Chapter 13 the approaches proposed to fill the gaps are explained and conclusively discussed in Chapter 14.

Lastly, in Part IV, Chapter 15 discusses the main highlights of this thesis and future perspectives.

In Part V, details about procedures and information about the benchmark methods are collected.

Chapter 2.

Background and Contributions

In this chapter, the general characteristics of NILM will be described. In detail, first the formulation of the problem is presented, and then the general NILM framework adopted in literature is presented. Datasets and the state-of-the-art will be deeply treated in the last sections.

2.1. Problem Statement

The total power measurement of a building can be modelled as the sum of all M power loads of the building plus noise $\epsilon(t)$, from measurement error and unknown loads:

$$y(t) = \sum_{k=1}^{K} s_k(t)x_k(t) + \epsilon(t), \tag{2.1}$$

where $s_k(t)$ is the state of activation and $x_k(t)$ is the power consumed by appliance k at the time instant t.

The problem can be approached mainly into two directions: by estimating $x_k(t)$, thus reconstructing the power consumption profile for each appliance of interest or by estimating $s_k(t)$, thus reconstructing the state of activation sample-by-sample for each appliance. Generally, a subset $\tilde{K} \leq K$ of appliances are in interest for the monitoring, based for example on their consumption or frequency usage. Thus, the problem can be reformulated as:

$$y(t) = \sum_{k=1}^{\tilde{K}} s_k(t)x_k(t) + \sum_{k=\tilde{K}+1}^{K} s_k(t)x_k(t) + \epsilon(t). \tag{2.2}$$

The second and third terms can be considered as noise because represent the undesired contribution. Thus, they generically can be expressed with:

$$v(t) = \sum_{k=\tilde{K}+1}^{K} s_k(t)x_k(t) + \epsilon(t). \tag{2.3}$$

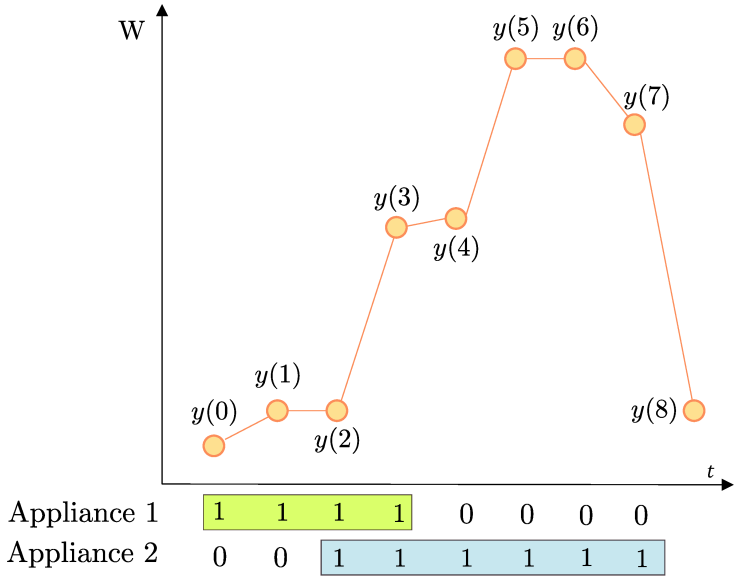


Figure 2.1.: Example of multi-label appliance classification task. For one window of aggregate power signal the aim is to localize in time for more than one appliance, where and which appliance is active.

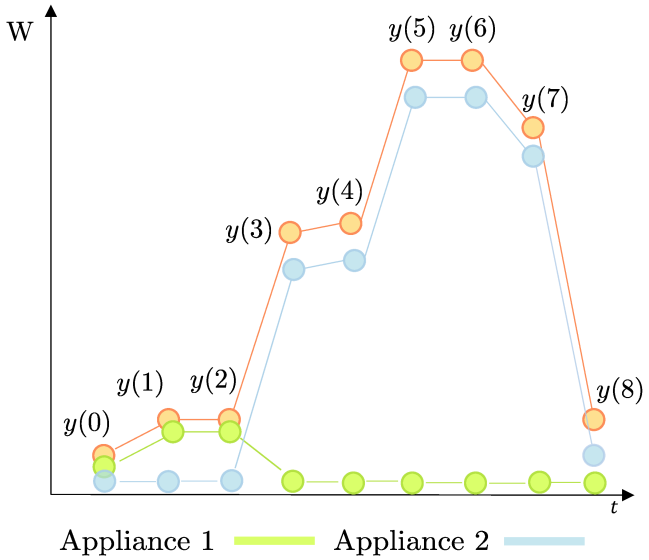


Figure 2.2.: Example of appliance power profile reconstruction task. For one window of aggregate power signal the aim is to estimate the power profiles of each appliance of interest.

Let $s_k(t)$ be the state that indicates if appliance k is ON at time sample t ($s_k(t) = 1$), i.e., if $x_k(t)$ is greater than a power threshold, or OFF ($s_k(t) = 0$). Then the classification task is to find $s_k(t) \in \{0, 1\}$, for all $k = 1, \ldots, \tilde{K}$ and $t = 1, \ldots, N$. A general version of the appliance classification is the multi-label appliance classification, where the task is to classify the state of multiple appliances for the same sample. The aggregate signal $y(t)$ can be divided into a series of J disjointed windows of size L samples where the j-th window is represented by the vector:

$$\mathbf{y}_j = [y(jL), \ldots, y(jL + L - 1)]^T \in \mathbb{R}^{L \times 1}. \tag{2.4}$$

Then, the corresponding series of J disjointed windows of states is:

$$\hat{\mathbf{S}}_j = [\hat{\mathbf{s}}(jL), \hat{\mathbf{s}}(jL + 1), \ldots, \hat{\mathbf{s}}(jL + L - 1)] \in \mathbb{R}^{\tilde{K} \times L}. \tag{2.5}$$

Note that above $\hat{\mathbf{s}}(t) = [\hat{s}_1(t), \ldots, \hat{s}_k(t)]$ is a predictions vector at the time instant t.

The appliance-power profile reconstruction consists in solving a regression task, where the output is a real value. According to recent published works, power consumption estimation is generally performed for each appliance separately. Thus, given the aggregate signal $y(t)$, the corresponding series of J disjointed windows of power samples is:

$$\hat{\mathbf{X}}_j = [\hat{\mathbf{x}}(jL), \hat{\mathbf{x}}(jL + 1), \ldots, \hat{\mathbf{x}}(jL + L - 1)]^T \in \mathbb{R}^{L \times 1}. \tag{2.6}$$

Figure 2.1 and 2.2 show two graphical examples for both classification and regression tasks for two appliances while Figure 2.3 shows a real example for both classification and regression desired output, represented for the same time period. The appliance monitored is the washing machine.

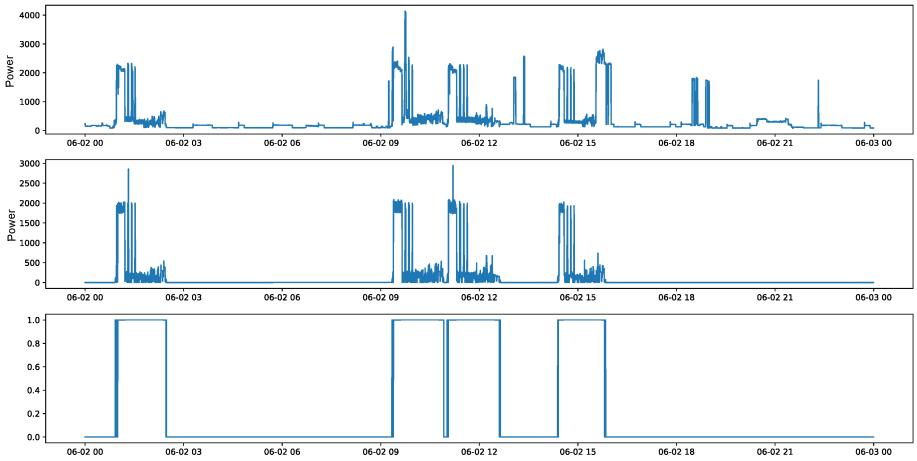


Figure 2.3.: Examples of output signals for regression (second row) and classification (third raw), for the same input window (first row).

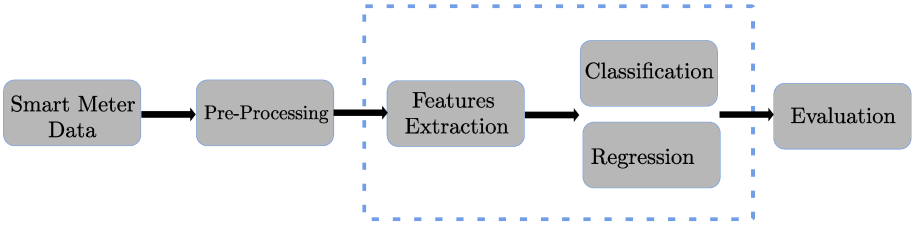


Figure 2.4.: NILM framework

2.2. General NILM framework

In Figure 2.4, the General NILM framework is shown. It is composed of the blocks "Smart Meter Data", "Pre-processing", "Features Extraction", "Classification", "Regression" and "Evaluation". According to the literature published until now, all NILM approaches follow these fundamental steps. What is different is principally how the blocks "Features Extraction" and "Classification"/"Regression" are modeled.

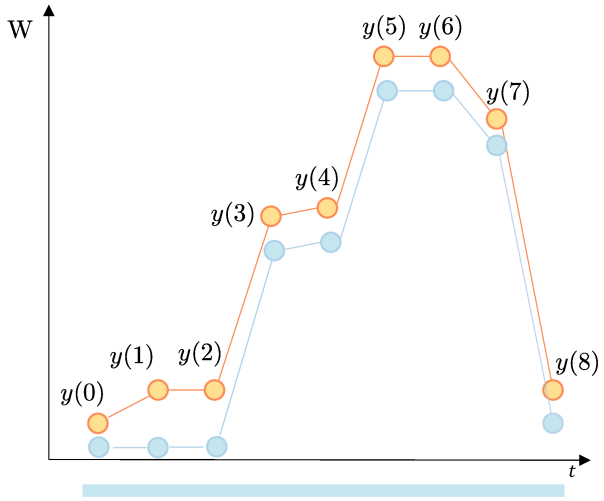
"Smart meter data" refers to the acquisition of active power consumption signals at the main meter. More generally, it can refer also to appliance-level sub-meters measurements, when a single meter for each appliance is applied at the device plug. The measurements are affected by errors or missing values due to sensor failures thus, "Pre-Processing" block refers to the phase in which signals are processed to fill the holes. Based on the gap entity, different strategies have been adopted [24, 25]. Signals can also necessitate to be down- or up-sampled based on the desired sampling frequency.

"Features extraction" consists in extracting high-level information from the

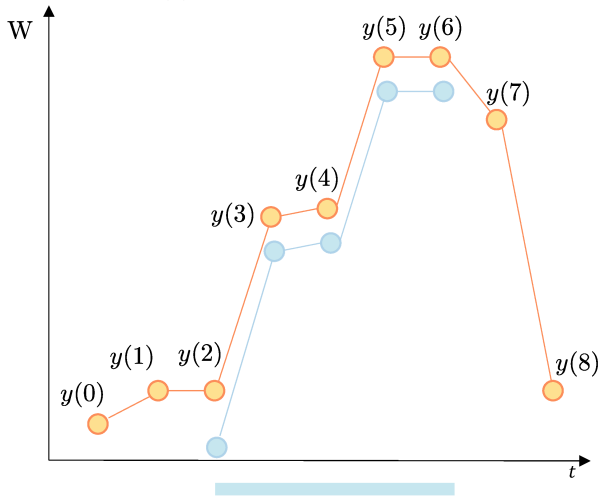
raw signals by applying specific statistics or automatically, as it happens for deep learning. Most common features are V-I trajectory (V: Voltage, I: Current), and Fast Fourier Transform (FFT). Other approaches used directly the raw power signals. "Classification" and "Regression" refer to the algorithms that perform the tasks and produce the predictions. In case of supervised or semi-supervised approaches, the availability of input data and the related desired outputs, generally called ground-truth, is strictly required for at least a part of the dataset. Considering the NILM tasks described in Section 2.1, for the regression task the ground-truth coincides with appliance power signal $\mathbf{X}_j$ and for the classification task coincides with the states of appliances $\mathbf{S}_j$ with values between 0 and 1. The input signal is the aggregate power consumption of the building. A real example is shown in Figure 2.3.

The relationship between input and output adopted until now in this chapter is the most general. The objective is always reconstructing sample-by-sample the desired output. What can be different is the way in which this is addressed, and there are mainly three models in the literature. The first is represented by the estimation of the desired disaggregated output with the same length L of the input window, for this reason called sequence-to-sequence. The second is the estimation of the central point of the desired output, also called sequence-to-point. The third is the estimation of a sub-sequence centred with the input window, also called sequence-to-subsequence. In Figure 2.5, all the three models are graphically reported. It is worth to clarify that the same holds for classification output $s_k(t)$.

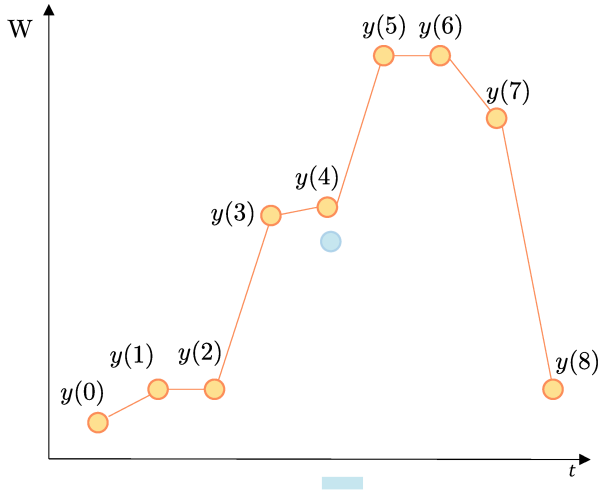
The last block, "Evaluation" refers to the phase in which the performance of the algorithms are evaluated on a test set. Common classification metrics are adopted to evaluate how much the network is able to detect the activation state. For the regression, some NILM-specific metrics have been proposed.



(a) Sequence-to-sequence.



(b) Sequence-to-subsequence.



(c) Sequence-to-point.

Figure 2.5.: Different input processing approaches. The signals are respectively the aggregate (orange) and appliance-level consumption (blue). The bar evidences the different lengths of the output sequence for the same input.

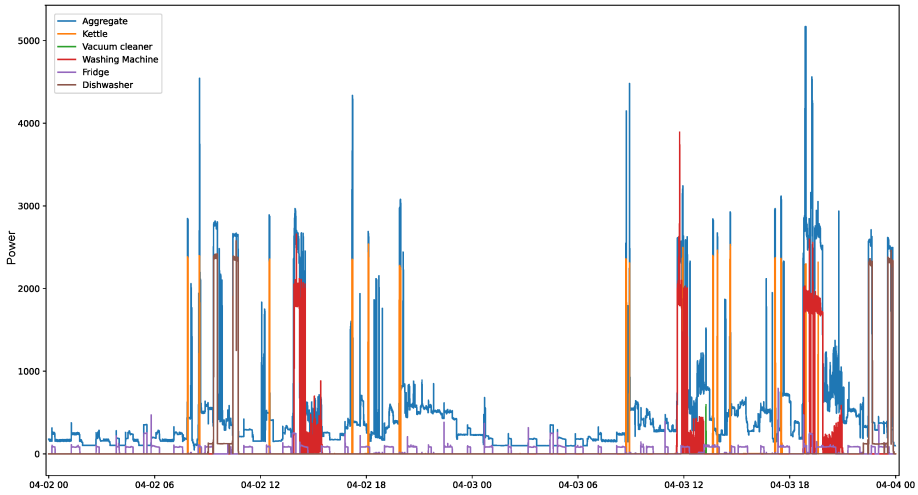


Figure 2.6.: Example of a daily consumption of House 1 from UK-DALE.

2.3. Datasets

NILM has become a widely investigated field of research, especially during recent years. The availability of numerous public datasets favored the research towards applying deep learning strategies that rely on large datasets to be effective. Public datasets promote fair comparisons because different works evaluate their algorithms on the same setup.

This section will go into details for the two of the most used low-frequency datasets during the last years (UK-DALE [24] and REFIT [25]) that will be used to develop and evaluate the proposed strategies. Additionally, other two of the most used public NILM datasets, REDD [26] and AMPds [27] are briefly described.

The UK-DALE dataset [24], introduced in the UK Domestic Appliance-Level Electricity study, is an open-access dataset comprising data from five residential houses. It has been publicly available since 2015. The dataset captures the active power consumption of individual appliances and the overall apparent power demand of each house. These measurements were taken every six seconds during a non-continuous period spanning from 2012 to 2015. In three of the houses, the voltage and current of the entire household were down-sampled at a rate of 1 Hz. Subsequently, the active power, apparent power, and root mean square (RMS) voltage were computed. For House 1, a total of 655 days of data are available, with individual recordings from nearly every appliance in the house. This results in a comprehensive dataset with 54 separate channels. In Figure 2.6 consumption signals of two days are reported for the House 1. The authors of the study also reported the percentage of sub-metered energy

for each house. House 1 accounts for 80% of the total aggregate consumption, while House 5 contributes 79%. House 2 follows with 68%, and Houses 3 and 4 have 19% and 28%, respectively. Upon analyzing the daily energy consumption patterns, it was found that certain appliances significantly contribute to the overall consumption. These key contributors include the kettle, dishwasher, home theater system, washer dryer, and fridge-freezer.

The REFIT Electrical Load Measurements dataset [25] has been publicly available since 2017. This dataset encompasses data from 20 residential homes over a continuous period of two years. Notably, all 20 homes were monitored at the same sampling rate. The data collection spans from September 2013 to July 2015. The dataset includes measurements of active power for both the household aggregate and individual appliances within each home. These measurements were recorded at 8-second intervals. During the monitoring period, the households carried out their typical domestic activities. The number of appliances installed in each house varies, ranging from a minimum of 15 to a maximum of 49. Among the represented appliances, the most common ones include the television, washing machine, microwave, kettle, dishwasher, and fridge.

In the Reference Energy Disaggregation Data Set (REDD) [26] the data is specifically geared toward the task of energy disaggregation, as declared by the authors. REDD consists of whole-home and circuit/device specific electricity consumption for a number of real houses over several months. For each monitored house are recorded: the whole home electricity signal (current monitors on both phases of power and a voltage monitor on one phase) recorded at a high frequency (15 kHz); up to 24 individual circuits in the home, each labeled with its category of appliance or appliances, recorded at 0.5 Hz; (3) up to 20 plug-level monitors in the home, recorded at 1 Hz, with a focus on logging electronics devices where multiple devices are grouped to a single circuit. The Almanac of Minutely Power dataset (AMPds) [27] contains one year of data that includes 11 electrical measurements at one minute intervals for 21 submeters. AMPds also includes natural gas and water consumption data. The authors specifically aimed their attention on the importance of having datasets that capture long-term usage of appliances. The AMPds dataset is a record of energy consumption of a single house using 21 sub-meters for an entire year (from April 1, 2012 to March 31, 2013) at one minute read intervals. They chose a one minute interval due to concerns over data communication network saturation, but this comes at a cost of loss of fidelity (i.e. missing power measurement spikes that could help identify loads more easily). The monitored house is in the region of British Columbia. Figure 2.7 shows an example of consumption signals for some appliances and the main.

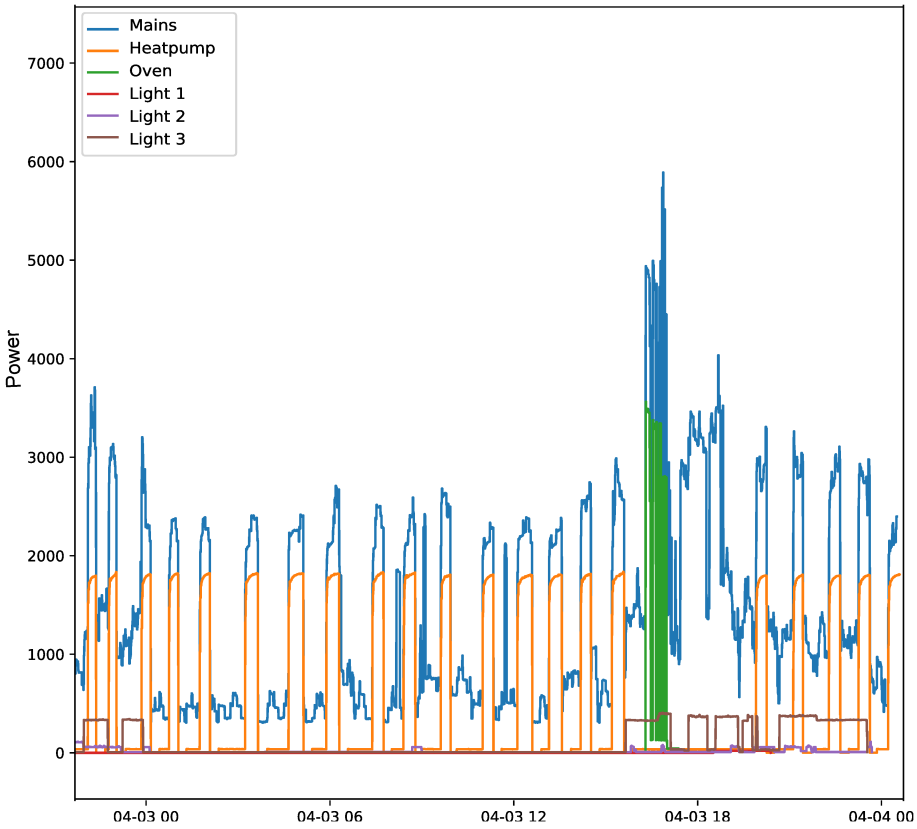


Figure 2.7.: Example of daily consumption from AMPds.

2.4. Related Works

Published NILM research has addressed classification and regression task equally. For both appliance-state classification and appliance power profile reconstruction, in this work only methods that exploited low-frequency power signals are considered.

Most of the published work have adopted a supervised learning strategy to address the NILM problem.

2.4.1. Power Profile Reconstruction

Among the supervised approaches for power profile reconstruction, Kelly et al. [28] firstly proposed three different architectures to estimate the power consumption of appliances from sequences of aggregate samples. The architectures were based on a de-noising Autoencoder (dAE), a Recurrent Neural Network (RNN), and the so-called Regress Start Time, End Time & Power network

composed of convolutional and fully connected layers. Similarly, in [29], two Convolutional Neural Networks (CNN) were trained with Root Mean Squared Error (RMSE) loss function. One network modeled the problem as sequence-to-point and the other the sequence-to-sequence approach. Kaselimi et al. [30] proposed an architecture with recurrent CNN units composed of two convolutional multi-channel modules. A Dilated-Residual Network has been proposed in [31] to prevent the vanishing gradient problem and training degradation, approaching the problem as sequence-to-sequence. Also recently, this architecture has been improved and proposed by [32]. Langevin and colleagues [33] used a Variational Auto-Encoder (VAE) to improve the disaggregation of power consumption of multi-state appliances and generalization performance. In [34], a method based on Generative Adversarial Networks (GANs) has been presented, where a dAE was trained using an adversarial training strategy, and a recurrent CNN units was employed as a discriminator. A Conditional-GAN approach was proposed in [35], where the problem was modeled as a sequence-to-subsequence estimation task. Self-Attentive-Energy-Disaggregation (SAED) from [36] has been developed to incorporate the attention mechanism into neural networks but mitigating the computational load. Ouzine et al. [37] proposed novel hybrid deep learning models providing the best disaggregation performances for multi-target disaggregation compared to single models. They explored both single- and multi-target models, with comparable performance. DiffNILM [38] is a recent published approach that, starting from random Gaussian noise, iteratively reconstructs the target waveform via a sampler conditioned on the total active power and encoded temporal features. An adaptive ensemble filtering framework integrated with long- and short-term memory (LSTM) is proposed for identifying flexible loads [39] such as heat pumps and electric vehicles. Recently, in [40], a deep learning model based on an attention mechanism, temporal pooling, residual connections, and transformers is proposed to effectively capture appliance-specific energy usage.

2.4.2. Appliance State Classification

Supervised approaches for appliances' states classification have been widely used as well. In [41] a CNN with temporal pooling is used to aggregate features of different time resolutions. Verma and colleagues [42] proposed a Multi-label Restricted Boltzmann Machine (ML-RBM) due to its effectiveness in learning high-level features and correlations. Singhal et al. [43] adopted Deep Dictionary learning to overcome low-frequency sampling-related problems and be more accurate with continuously varying appliances. Singh et al. [44] adopted a Sparse Representation Classification (SRC) while reducing the number of data collected for training. Massidda et al. [45] implemented temporal pooling to

concatenate different time resolution information. A recurrent network structure is adopted by Cimen et al. [46] that proposed a Gated Recurrent Units (GRUs) based approach, where features from the aggregate signal and spikes are extracted before by using convolutional layers. Zhuo et al. [47] proposed a convolutional-recurrent and random-forest (RF) based architecture to address label correlation and class-imbalance problems. Dealing with the time-varying nature of power signals, Verma et al. [48] proposed an encoder-decoder architecture based on a Long Short-Term Memory network (LSTM) to model complex dynamics. In [49], a CNN followed by three different fully connected sub-networks was implemented for multi-label state and event type classification. Deep Blind Compressed Sensing has been proposed by Singh et al. [50], exploiting compressed information to reduce transmission rate to detect devices' states. In [51], authors proposed a multi-objective approach where one model is used for many appliances, incorporating the appliance transfer learning.

2.4.3. Multi-task approaches

Some works argued that combining the two tasks could lead to mutual benefits. Thus, multi-task architectures have been proposed generally by using double-branched architectures, to perform both classification and disaggregation, exploiting the correlation between the two tasks. Murray et al. [52] proposed two architectures, one CNN-based and the other based on Gated-Recurrent Units (GRU). Both networks were composed of two branches, one for classification and the other for disaggregation. Piccialli and Sudoso [53] also proposed a dual tasks architecture where the regression sub-network was improved with an attention layer, and the regression output was combined with the related classification prediction. Liu and colleagues [54] proposed the so-called SAM-Net, a scale- and attention-experts based multi-task neural network to make full use of the correlation between the tasks of the NILM.

2.5. Open Issues and Contributions

The research conducted so far has successfully determined the status of appliances and recreated their power profiles. However, to implement these methods in real-world situations, several obstacles need to be addressed.

These studies typically presume the availability of a substantial amount of labeled data for network training. In practice, this is only possible if there is a person who manually annotates the on and off times for each appliance activation, specifically for state classification of appliances. For power profile reconstruction, the alternative is to install sensors for each device, which can

be impractical and costly.

Another common characteristic of supervised methods is the use of large, multi-layered architectures to enhance feature capture and learning. Consequently, these networks have millions of parameters that need to be trained, requiring powerful platforms for training and deployment. Given their computational intensity, it is assumed that the computation will occur in the cloud. This could potentially lead to privacy and latency issues, resulting in a less than optimal service for the user.

Then, the contributions of this work are the following:

1. Bridging the gap between Non-Intrusive Load Monitoring (NILM) methods and the user's role, which is crucial in data annotation. So far, the user's active participation has been only slightly considered.

2. Incorporating the user into the development process of NILM algorithms to take advantage of various annotations. This approach can lighten the labeling effort required from the user.

3. Simplifying NILM algorithms and reducing storage requirements to operate closer to the end-user. This way, at least the inference can be performed on devices with limited resources.

The following chapter will describe the overall framework proposed in this work to fill the research gaps, by introducing the deep learning methodologies adopted to develop NILM strategies.

Chapter 3.

Edge-Centric Non-Intrusive Load Monitoring Framework

This chapter describes in detail the approach adopted in this work to address the various open challenges that NILM approaches should deal with. For this reason, as anticipated in Chapter 1, the core of the framework leverages techniques associated with Deep Learning and the paradigms of Human-Centred Computing and Edge Computing. By the use of well-known deep learning strategies, this work aims to drive NILM towards Edge-Centric characteristics and applications.

As depicted in Figure 3.1, Deep Learning and the two paradigms will be embedded together in developing the NILM methodologies that will be presented in the following sections. It is worth to clarify that not all the deep learning techniques associated with the paradigms directly belong to them. The purpose is to describe that by developing methods that follow such techniques, NILM will be closer to meet paradigms' principles. For example, transfer learning is not by itself a technique related to a human-centric vision, but by using that strategy linked to weak supervision it has been possible to better reduce the labeling effort in favor of the user and to approach the Human-Centred paradigm closer. Next sections will describe firstly the deep neural network adopted to develop the NILM methods presented in this thesis. Then, the deep learning techniques related to the Human-Centred Computing and Edge Computing paradigms adopted in this work will be exposed.

3.1. Convolutional Recurrent Neural Network

In literature, Deep Neural Networks (DNNs) have been structured in diverse ways to perform a variety of tasks. For instance, tasks related to computer vision and images typically employ convolutional networks. Conversely, recurrent structures are predominantly used with time series data.

In conventional Convolutional Neural Networks (CNNs), convolution and pooling operations are performed independently for distinct regions of an im-

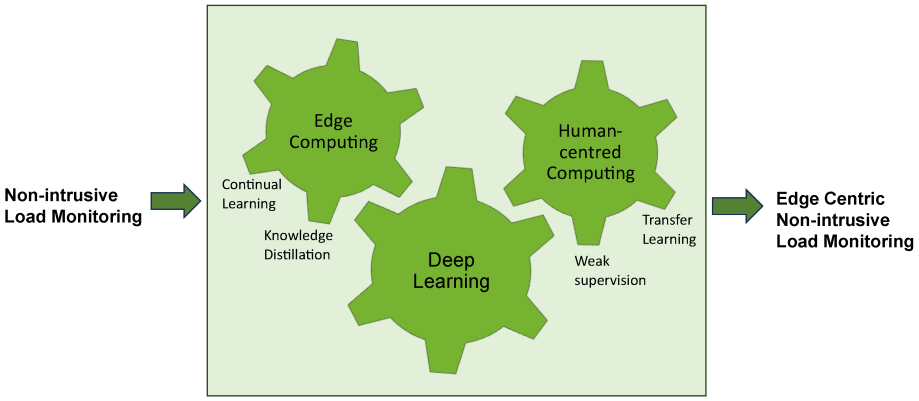


Figure 3.1.: Overall approach adopted for Edge-Centric Non-Intrusive Load Monitoring.

age. However, these methods overlook the contextual relationships that exist among these regions. Recognizing these relationships can offer significant insights into the structure of images.

Conversely, Recurrent Neural Networks (RNNs) are specifically engineered to understand these contextual relationships within sequential data, courtesy of their recurrent connections. A Gated Recurrent Unit (GRU) is a variant of RNN that is employed for sequence-to-sequence challenges, such as language translation or speech recognition.

In this thesis, a hybrid model known as the Convolutional Recurrent Neural Network (CRNN) [55, 56] is utilized. This model is crafted to comprehend local features extracted by the convolutional layers and global behaviors through the recurrent component.

As illustrated in Figure 3.2, the Convolutional Recurrent Neural Network (CRNN) consists of three primary components. The first component includes several blocks, each containing a 1D or 2D convolutional layer with filters and a kernel of a specific size, a batch normalization layer, an activation layer with a Rectified Linear Unit (ReLU) function, and dropout for regularization.

Batch normalization [57] is generally used to expedite training by normalizing the input layer, which helps stabilize the network during training. The ReLU function introduces non-linearity to the deep learning model, which helps mitigate the vanishing gradient problem. Essentially, it processes the input and outputs the value if positive; otherwise, it outputs zero. This function is widely used in deep learning due to its effectiveness.

Dropout refers to the technique of randomly dropping out nodes in a neural network during training, which helps prevent over-fitting by introducing random noise. Typically, a pooling layer is placed between the activation and

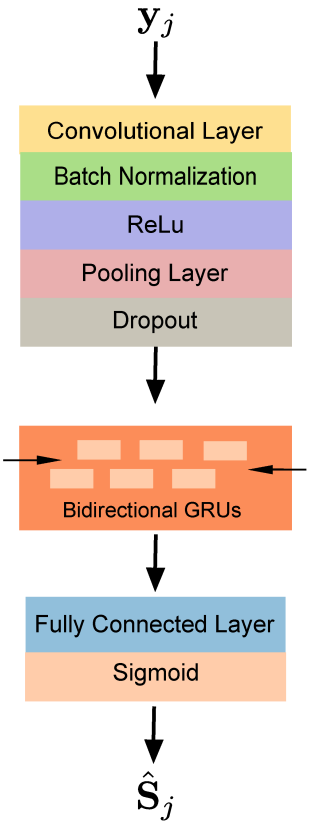


Figure 3.2.: General CRNN architecture. For the sake of conciseness, only one convolutional block is graphically reported. The example regards the multi-label classification task, thus the sigmoid function is adopted in the fully connected layer to produce the outputs.

dropout layers to perform sub-sampling. However, in this work, the pooling layer is not included in the convolutional block. This is because the input and output windows have the same length, making sub-sampling unnecessary.

The convolutional block's output is channeled into the network's recurrent section, which consists of a bidirectional layer of multiple Gated Recurrent Units (GRUs) [58]. The inclusion of a bidirectional recurrent layer is motivated by the need to incorporate both future and past timestamps. This allows the sequence to be processed from start to end and vice versa, which is beneficial for tasks where context is crucial for accurate prediction.

The network's final section is a fully-connected layer, followed by a sigmoid activation function used for binary classification. In the case of regression, a linear function is used for activation. This architecture is particularly suited for Non-Intrusive Load Monitoring (NILM), as the convolutional section extracts features related to the appliance load signature, while the recurrent section extracts temporal information related to activation and context.

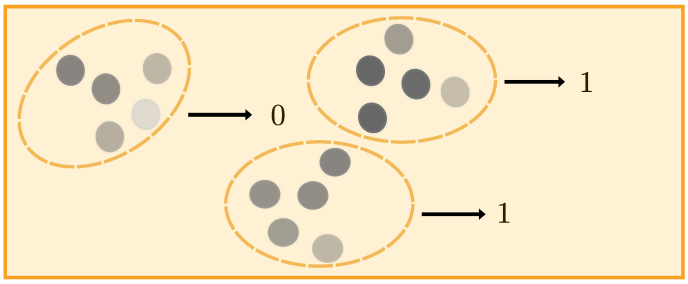
CRNNs have been previously applied to other application domains [59, 60, 61, 62]. Nonetheless, this architecture has never been employed before for multi-label appliance classification, to the best of authors knowledge.

3.2. Weak Supervision

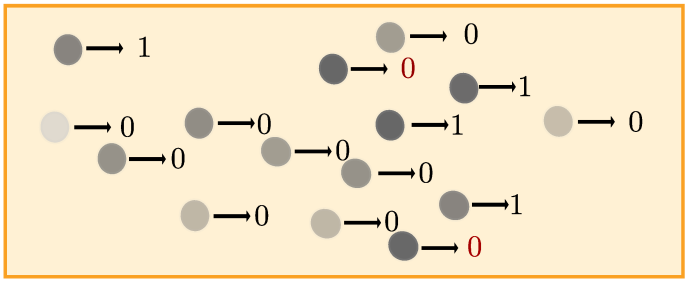
Weak Supervision refers to collection of learning strategies that exploits data partially labeled for training supervised deep learning models.These strategies vary depending on the nature of the missing information, as illustrated in Figure 3.3.

Inexact Supervision refers to a type of supervision that provides information for all the data but partial supervision is given, not fine-grained as expected. The content is coarser compared to information used by supervised approaches. Generally the coarse-grained label is assigned to a group of so-called "instances" that share similar characteristics but are different. The group of instances is called "bag". This type of learning is also called Multiple Instance Learning (MIL), since the network learns from one information related to multiple instances at the same time. Significant research effort is directed to properly model the best relation between instance-level and bag-level labels. This type of supervision is largely applied for sound event detection [63, 64, 61] and image recognition [65, 66].

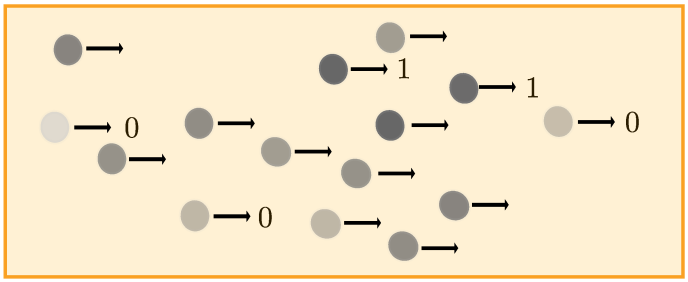
Another type of weak supervision is Inaccurate Supervision. In this case, labels assigned to data points might be incorrect or contain errors, deviating from the true ground truth. In other words, some labels may suffer from errors. In this case, strategies to identify errors and try to correct them are open research topics.



(a) Inexact Supervision.



(b) Inaccurate Supervision.



(c) Incomplete Supervision.

Figure 3.3.: Weak Supervision. A binary classification example is represented.

Lastly, Incomplete Supervision refers to the availability of a small labeled set and a larger set of unlabeled data. By itself the small labeled set is not sufficient to be used in training. To solve this scenario, active learning [67] and semi-supervised learning [68] are widely researched techniques. The difference among them is the way in which missing labels are obtained. The first assumes that there is an "oracle", generally a human expert, that can be queried to get ground-truth labels for properly selected unlabeled instances. In contrast, semi-supervised learning automatically exploits unlabeled data in addition to labeled data to improve learning performance.

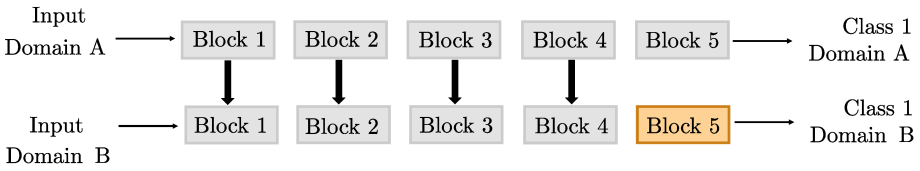


Figure 3.4.: Schematic representation for transfer learning. The example refers to the case when the task to perform is the same but data belong to different domains. All the blocks that learnt common low-level features about the task are transferred to the new network and only the last layer is fine-tuned on the new domain data.

3.3. Transfer Learning

Transfer learning is a well-known technique and consists in leveraging feature representations from a pre-trained model, without training a new model from scratch when applied on a different scenario. A different scenario can be characterized by new data domain or new tasks. A recent review on this technique has been published by [69].

New tasks generally share some characteristics with previous tasks learnt during training. One basic example can be a network trained to classify dogs' images, then used to classify cats. The new data domain refers to evident differences among data used for training and the data to be processed in the operative phase. When moving to a new task or data domain, especially for supervised trained networks, issues related to low performance have been reported. It can happen that network weights over-fit on training data and the network is not able to correctly process slightly different data. Another case is when the network can discreetly generalize on new data data but they are intrinsically different to the ones used in training. In NILM, domain differences are very frequent and generally are related to household appliances, frequency usage, appliances types and operational modes. Thus, to overcome this issue, a common transfer learning technique called fine-tuning is required. Fine-tuning consists in freezing the layers that learnt the low-level features (common among different domains or different tasks) and re-training only the last layers of the network to incorporate knowledge about new data or new tasks. In Figure 3.4, fine-tuning is represented. Grey blocks refer to pre-trained layers on domain A. Except for the last layer, the weights are fixed also to perform task B and only the orange block is trained to fit with domain B data. The same holds for different tasks.

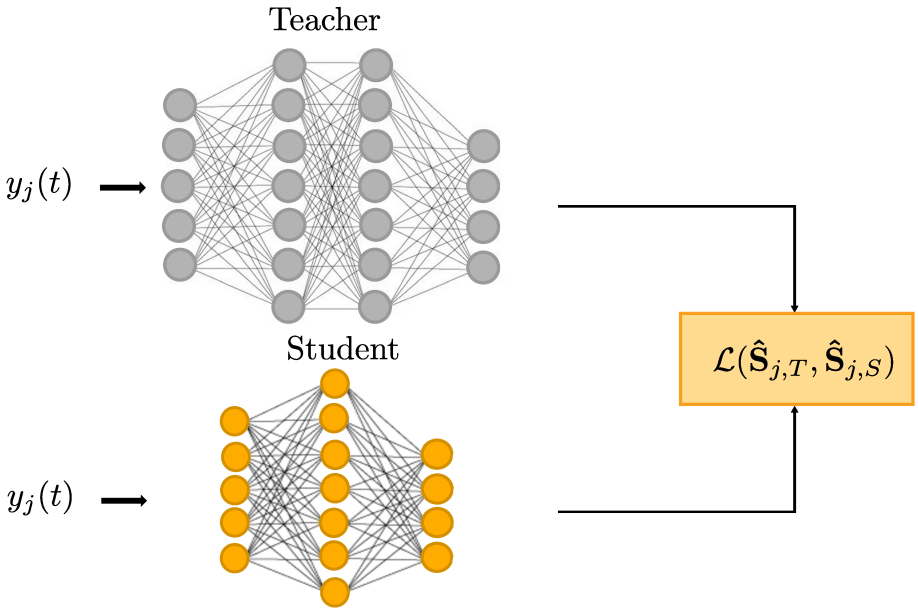


Figure 3.5.: Knowledge Distillation based on Teacher-Student strategy for a model compression application. The $\mathcal{L}$ function is a generic loss function.

3.4. Knowledge Distillation

Knowledge Distillation (KD) has been firstly proposed by Hinton et al. [70], inspired by mechanism that enables humans to quickly learn new complex concepts when given only small training sets with the same or different categories. The mechanism has been adopted to (i) transfer the knowledge from one domain to another and (ii) model compression. Most of the works applied this principle by using the so-called Teacher-Student architecture. Generally, when the objective is transferring the knowledge, the structure of Teacher and Student networks is the same. For model compression, the Teacher consists in a larger network with high number of parameters and the Student is a smaller network. Sometimes, the type of the Student network is different [71]. Consider the example in Figure 3.5 where the networks are both Multi-layer perceptron (MLP) and in the Student network one hidden layer has been removed. The input and output layers have the same dimension. The purpose is to distill the knowledge from a Teacher model, ideally one performing close to optimally, to the Student model with significantly fewer parameters. In this way, the Student network tries to mimic the Teacher in performance. It can happen that reducing the number of parameters favors a slight improvement of performance, probably due to removal of redundant parameters. Large networks tend

to over-fit, especially if the quantity of training data is not balanced with the architecture. In the approach initially proposed by Hinton [70], the distillation mechanism occurs through a loss term, where the Teacher outputs are used as labels from which the Student can learn. Knowledge distillation is also used to leverage previously acquired knowledge on the source data domain, when data are not sufficient to learn a new target task. In this way, smaller Student network can better fit with the available data, preventing under-fitting.

3.5. Continual Learning

Continual Learning, also known as Incremental Learning or Life-long Learning, is a research branch that aims to move beyond the static knowledge paradigm of deep learning. Its goal is to enable models to correctly process new instances, new classes, or new tasks without suffering from catastrophic forgetting and adapting to new real-world dynamics. Forgetting phenomenon occurs when a model's performance on old tasks degrades significantly after learning new information. This can happen when completely re-training the network to extend the performed tasks. As clearly highlighted in [72], in a dynamic world, this practice quickly becomes intractable for data streams or may only be available temporarily due to storage constraints or privacy issues. This calls for systems that adapt continually and keep on learning over time. To address this challenge, researchers have proposed various techniques falling into three main categories: architectural or parameter isolation [73, 74, 75], rehearsal [76, 77], and regularization [78, 79, 80] strategies.

Architectural strategies aim to solve catastrophic forgetting by designing neural network architectures that mitigate catastrophic forgetting. For instance, Progress neural network (PNN) [73] and Expert Gate [74] assign one neural network column to each task, ensuring that the parameters of previous tasks are fixed. One approach that considers scalability issues is Packnet [75]. Inspired by network pruning techniques, it optimizes large deep networks by freeing up parameters for new tasks while maintaining performance and minimizing storage overhead, sequentially "packing" multiple tasks into a single network.

Rehearsal techniques save exemplar data from previous tasks to combat forgetting. Experience Replay [76] replays the exemplar set while learning the current task and Gradient Episodic Memory [77] which computes gradients in the exemplar set for each task, ensuring they align with the direction of gradients from previous tasks.

Regularization strategies regularize the forgetting on previous tasks by loss. Learning without Forgetting [78] uses distillation loss to maintain the output distribution between old and new tasks. Elastic Weight Consolidation [79] and Synaptic Intelligence [80] impose a higher penalty on precise parameters for

previous tasks, causing the optimized trace to follow non-precise parameters to optimize previous tasks and current tasks.

Continual learning is crucial for AI deep learning-based systems to adaptively develop knowledge over their lifetime, and understanding these strategies helps address the challenges posed by real-world dynamics. For more details, refer to the recently published review by DeLange et al. [72], where a comprehensive comparison of the common approaches is reported.

Part II.

User-centred NILM Methods

The first and second contributions of this work, listed in Chapter 2, will be deeply illustrated and discussed in this part of the thesis. In Chapter 4, an introduction to the necessity of reducing the annotation effort in NILM applications will be presented. Accordingly, the approaches already proposed in literature to address this problem are described.

To lighten the role of the users in labeling data, NILM has been modeled as a Multiple Instance Learning and the Multiple Instance Regression problem in Chapter 5 and Chapter 6. In this way, a coarser label can be obtained by the user feedback and exploited to train the network.

Once demonstrated the efficacy of this learning approach and the possibility to improve the performance, a transfer learning approach is proposed in Chapter 7 to pose the NILM framework in a more real scenario, where large differences between training and target data are commonly evident. Then, merging the weakly supervised transfer learning approach and an advanced data selection strategy based on active learning, the quantity of data to be annotated are further reduced in Chapter 8.

Overall results are discussed in Chapter 9. Adopting weak supervision and then an efficient data selection based on weak predictions, the annotation effort is consistently reduced by progressively increasing the performance.

Chapter 4.

Introduction

A drawback of strongly supervised methods, presented in Section 2.4, is that they require large amounts of labeled data for training the networks. Especially in practical scenarios, it is hard to obtain.

Alternative approaches have been developed to handle scarcity of annotated data, as illustrated in Section 3.2. Specifically for NILM, semi-supervised approaches are able to exploit unlabeled data, thus they require fewer annotations to achieve similar state-of-the-art performance [5, 81]. However, the number of published papers on these semi-supervised methods is less than that of supervised methods.

SARAA [82] is a semi-supervised learning process for automated residential appliance annotation that produced classifiers with performance 14.8% lower than the benchmark classifiers trained on the fully labeled ground truth data. In a later work, Yang and colleagues [5] proposed a Teacher-Student architecture based on Temporal Convolutional Networks (TCN) for multi-label appliance classification to exploit unlabeled data and include them in the training process. In [81], Virtual Adversarial Training (VAT) was used for energy disaggregation to train a sequence-to-point network. Learning was based on a regularization term calculated as the average of local distributional smoothness (LDS), and superior performance was obtained compared to fully supervised learning.

As treated in Section 3.2, the lack of labels can be managed also by annotating only the most significant available data. In this way, the selection of a noteworthy set excludes the use of unlabeled data and strengthen the training with significant data. This strategy is known as the Active Learning (AL) [83] that is a methodology designed to minimize the labeling effort required to train Deep Learning (DL) algorithms. The process consists in selecting only a subset of data for which the ground-truth is required while keeping an acceptable level of performance. Unlabelled data samples belonging to the so-called "query pool" are usually ranked according to informativeness, distance criteria, or a combination of both. Labels are then requested only for the data samples that are expected to contribute most significantly to the model’s training.

In the active learning scenario, the figure of annotator is identified in a human expert. In NILM applications, the human can be the user. But it is very unlikely that a user is skilled about the activation cycle of the dishwasher or the washing machine to properly annotate it. Thus, another skilled figure should be introduced or the user should be involved differently. AL for NILM has not been extensively investigated yet - there have only been a few attempts for event-based methods using high-frequency load measurements, based on: k-Nearest Neighbours (k-NN) in [84], Support Vector Machines (SVM) in [85], Random Forest with semi-supervised and AL combined in [86], and a DNN, using high-frequency measurements and event detection in [87]. Only one approached the problem by using low-frequency measurements and supervised model-based NILM in [88]. A supervised AL-based framework was proposed [88] to find the trade-off between accuracy and number of queries to enlarge the training set in an unseen domain and to improve the transferability of NILM models. Although improving performance, this approach relied on a small original training set with strong labels, necessitating sample-by-sample annotations. It remains the only approach that uses low-frequency measurements and supervised model-based NILM [88].

Both semi-supervision and active learning rely on ground-truth information, at least for a subset of the dataset. However, none of the previous low-frequency works that adopted these strategies have in-depth considered the role of the end-user as an annotator, nor the implications of involving them in the annotation process. The study by Rossier et al. [20] briefly involves the user in the process of turning appliances on and off as needed for system training and label acquisition. This method, however, is notably labor-intensive and time-consuming. Conversely, the study by Berges et al. [21] incorporates user intervention in identifying appliance signatures to cut down on the costs of sensor installation. This approach necessitates that the user possesses technical skills. Thus, from a practical point of view, they still face issues related to obtaining ground-truth information in real-world applications. A large effort can discourage the user from participating, and, also, providing precise timing information about appliance usage is more prone to errors.

In this part of the thesis, the first and second contributions of this work listed in Section 2.5 will be deeply illustrated and discussed. A novel method based on inexact supervision is proposed. NILM is modelled as a Multiple-Instance Learning problem where labels are assigned to an ensemble of aggregate samples. The end-user can more easily provide this information. Firstly, the general approach based on Multiple-Instance Learning is proven effective, comparing it with two state-of-art works in Chapter 5. In Chapter 6, the method is appropriately modeled to perfom appliance profile reconstruction. Then the learning method is extended transfer learning scenario to exploit less labeled

data, reproducing a more practical case in Chapter 7. Lastly, weak supervision is embedded with active learning in Chapter 8. In this way, the role of end-user as annotator will be lighten even more both from the point of view of (i) ground-truth information requested and (ii) quantity of data to be annotated. The weakly supervised active learning provides only the meaningful data to be annotated.

Chapter 5.

Multi-Label Appliance Classification with Weakly Labeled Data

This chapter will provide a comprehensive overview of the method for multi-label appliance classification, which has been modeled as a Multiple Instance Learning (MIL) problem. The method has been published in [2].

The concepts of *instances* and *bags* have already been introduced in Section 3.2. Thus, it is worth defining *strong* labels and specifically *weak* labels, that will be used in this work to reduce the labeling effort. *Strong* labels are labels assigned sample-by-sample, thus to each instance (sample) of the signal. These labels are the commonly used in supervised approaches. On the other hand, the ensemble of the instances is represented by the bag, to which it is possible to assign the weak labels. Bag labels are noisy, coarse, and inexact, thus they are commonly referred to as *weak labels*.

Both labels type will be employed in the learning process of a deep neural network, trained to identify the state of multiple appliances sample-by-sample. This solution has the dual consequence of improving generalization capability compared to supervised approaches and reducing labeling costs. In principle, appliance classification does not necessarily require active power signals of individual appliances for training since annotation can be performed manually. Thus, sensors' expenses are not mandatory and as a consequence, the metering infrastructure can be simplified. On the other hand, manual annotation to provide strong labels requires a significant human effort that would not be easy to afford. With weak labels, manual annotations are provided on a wide temporal window, thus, it is sufficient to indicate if an appliance was active or not within that segment by using only a single weak label. In this sense, the method can also deal with the inexactness that may originate from mislabeling by manual annotators.

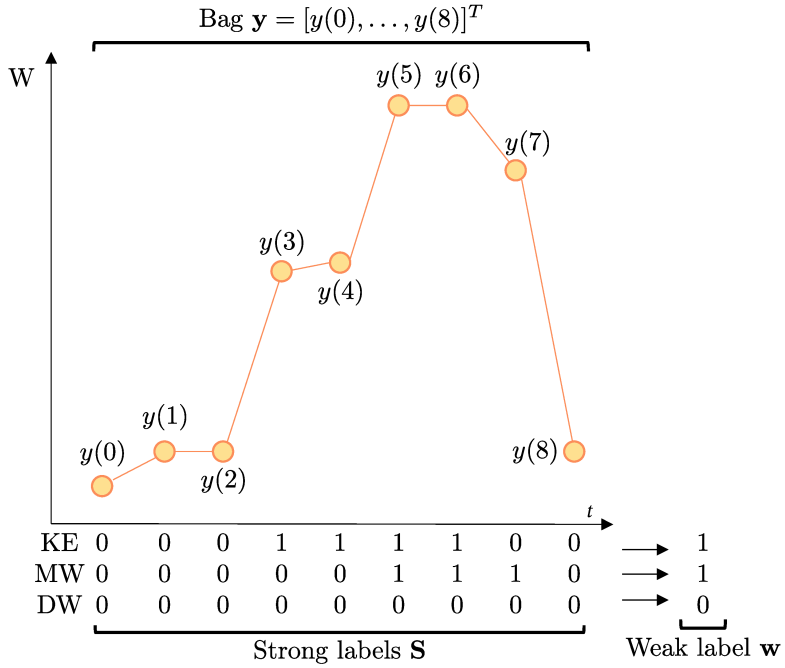


Figure 5.1.: Schematic representation NILM formulated as Multiple-Instance Learning. KE: Kettle. MW: Microwave. DW: Dishwasher.

5.1. Proposed methodology

MIL is a different form of supervised learning and a particular variant of weak supervision [89]. In MIL, learning examples are represented by *bags* composed of multiple *instances* (e.g., feature vectors, raw samples), and labels are provided only at the bag level. During prediction, the objective can be to classify bags, individual instances, or both [90]. MIL can be applied to single-label classification tasks, where only one label is assigned to bags and instances, or to multi-label classification tasks, where labels are multiple (multi-instance multi-label learning, MIML) [91]. Labels assigned to bags depend on the labels of individual instances inside them. In binary classification tasks, the *standard multiple instance assumption* states that the necessary and sufficient condition for a bag to be assigned a positive label is that one of its instances is positive, but later works have proposed other alternatives [92]. The same criterion can be easily extended to multi-class problems.

In the proposed method, instances are represented by the raw samples of the aggregate signal $y(t)$, and the related labels are represented by one-hot vectors $\mathbf{s}(t) \in \mathbb{R}^{\tilde{K} \times 1}$ defined as:

$$\mathbf{s}(t) = [s_1(t), s_2(t), \ldots, s_{\tilde{K}}(t)]^T. \tag{5.1}$$

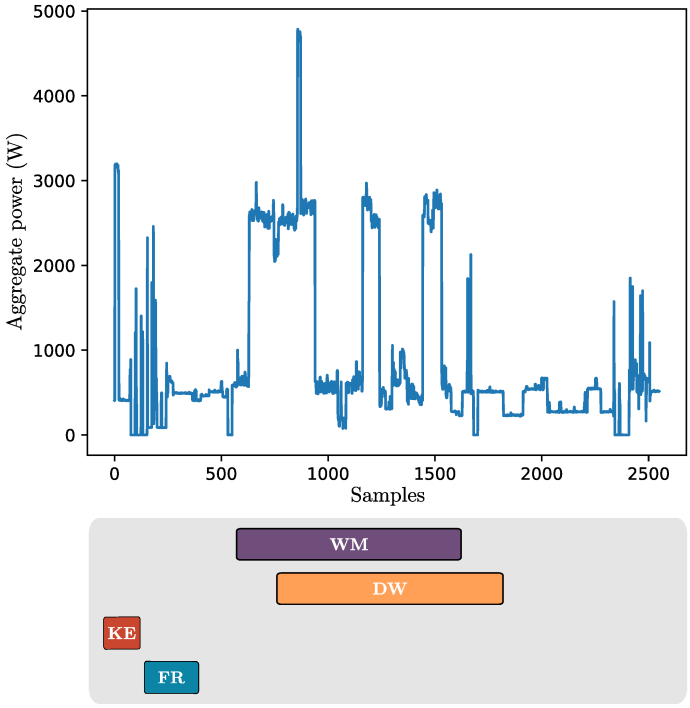


Figure 5.2.: An example of aggregate segment from house 2 of REFIT with the related labels. The weak label is represented by the presence of the tag with the appliance name, meaning that inside the window the appliance is active at least one time. The dimension of the coloured segment defines the ON- and OFF-time of the appliance activation. KE, MW, FR, WM, and DW stand respectively for Kettle, Microwave, Fridge, Washing Machine, Dishwasher.

A bag is a segment of $y(t)$ with length L. Supposing that $y(t)$ is divided into disjointed segments, the j-th bag is represented by the following vector:

$$\mathbf{y}_j = [y(jL), \ldots, y(jL + L - 1)]^T \in \mathbb{R}^{L \times 1}. \tag{5.2}$$

The related label is again encoded as a one-hot vector $\mathbf{w}_j \in \mathbb{R}^{\tilde{K} \times 1}$. As aforementioned, $\mathbf{w}_j$ depends on the instance labels inside it.

In Figure 5.1, a schematic example of instances and bag for NILM is reported. For each sample of the aggregate signal, strong labels represent the state of multiple appliances. The bag is the ensemble of these samples and the bag-level label (weak label) depends on the strong labels related to instances. The *bag* level, thus, contains information on the presence of one or more appliances in a time window, while, at the *instance* level, this information is provided at sample resolution. To clarify the concept, in Figure 5.2 an example of a

bag of active power aggregate consumption is reported. Through weak labels associated to the bag it is possible to know which appliances are active inside the window. On the other hand, to localize the activation inside the window, strong labels are necessary.

Denoting with $\mathbf{S}_j = [\mathbf{s}(jL), \mathbf{s}(jL+1), \ldots, \mathbf{s}(jL+L-1)] \in \mathbb{R}^{\tilde{K} \times L}$ the set of instance labels related to segment j, the relationship can be represented by a pooling function $\mathbf{b} : \mathbb{R}^{\tilde{K} \times L} \to \mathbb{R}^{\tilde{K}}$ such that

$$\mathbf{w}_j = \mathbf{b}(\mathbf{S}_j). \tag{5.3}$$

Several pooling functions have been proposed in the literature, each having different characteristics [60]. The pooling function used in this work will be defined in the following section, along with the neural network architecture.

In this work, the objective is to identify if an appliance is active or not at the sample level, thus the goal is to learn a function $\mathbf{f} : \mathbb{R}^{L} \to \mathbb{R}^{\tilde{K} \times L}$ such that:

$$\hat{\mathbf{S}} = \mathbf{f}(\mathbf{y}), \tag{5.4}$$

where $\mathbf{y}$ is an unknown aggregate segment, and $\hat{\mathbf{S}}$ contains the estimated instance-level probabilities for each class. The bag index j has been omitted for simplicity.

5.1.1. Neural Network Architecture

The function $\mathbf{f}(\cdot)$ in Equation 5.4 is represented by a CRNN, already presented in Section 3.1. The block scheme specifically related to this work is depicted in Figure 5.3. For each segment $\mathbf{y}_j$, the network produces the related instance-level estimate $\hat{\mathbf{S}}_j$ and the bag-level estimate $\hat{\mathbf{w}}_j$ for each appliance in interest. Convolutional blocks are identified with H, with F filters and kernel of size K_e, while the recurrent part comprises U Gated Recurrent Units. The final layer is denoted as *instance* layer, and it produces the instance-level estimate $\hat{\mathbf{S}}_j$.

After the instance layer, a pooling layer followed by a sigmoid activation function produces the bag-level prediction $\hat{\mathbf{w}}_j$. The proposed network, thus, has both an instance-level output and a bag-level output. In this way, it is possible to conjugate MIL with the supervised learning strategy based on strong labels. The pooling layer implements the pooling function $\mathbf{b}(\cdot)$. As aforementioned, several alternatives exist for the pooling function. Based on the analysis conducted in [60], the *linear softmax* is chosen since it achieved the highest localization performance. The linear softmax pooling function calculates the k-th

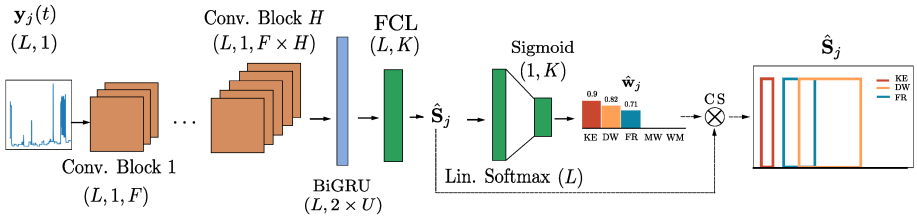


Figure 5.3.: Block scheme of the proposed approach. The network takes as input the aggregate power related to the j-th window and produces two outputs, one from the instance layer ($\hat{\mathbf{S}}_j$) and one from the bag layer ($\hat{\mathbf{w}}_j$). The multiplication is related to clip smoothing. T is length of input and output window, FCL stands for Fully Connected Layer, K is number of appliances.

element of $\hat{\mathbf{w}}_j$ of the bag-level prediction as:

$$\hat{w}_{k,j} = \frac{\sum_{t=jL}^{L(j+1)-1} \hat{s}_k^2(t)}{\sum_{t=jL}^{L(j+1)-1} \hat{s}_k(t)}. \tag{5.5}$$

In this way, the larger instance-level predictions receive a larger weight [60].

5.1.2. Learning

Learning from bags raises important challenges that are unique to MIL formulation [90]. As aforementioned, from a single weak label, multiple combinations of instances exist that can produce the same bag label, thus it is expected that a learning algorithm trained only on weakly annotated data achieves inferior results than training on strongly annotated data. The availability of several datasets for NILM with strong annotations [93] motivated us to train the CRNN by using both weak and strong labels.

More in detail, denoting with $\mathcal{T}_w = \{(\mathbf{y}_1, \mathbf{w}_1), \ldots, (\mathbf{y}_{M_w}, \mathbf{w}_{M_w})\}$ the set of training bags annotated with weak labels and with

$\mathcal{T}_s = \{(\mathbf{y}_1, \mathbf{w}_1, \mathbf{S}_1), \ldots, (\mathbf{y}_{M_s}, \mathbf{w}_{M_s}, \mathbf{S}_{M_s})\}$ the set of training bags annotated with strong and weak labels, learning is performed on the training set $\mathcal{T} = \mathcal{T}_w \bigcup \mathcal{T}_s$.

The loss function is composed of the weighted sum of the binary cross-entropy losses calculated on strong and weak labels:

$$\mathcal{L} = \mathcal{L}_s + \lambda \mathcal{L}_w, \tag{5.6}$$

where $\mathcal{L}_s$ and $\mathcal{L}_w$ are respectively the loss related to strongly and weakly labeled data, and the weight λ balances their contribution.

The two loss terms $\mathcal{L}_s$ and $\mathcal{L}_w$ are the Binary Cross-Entropy (BCE) for each

class and they are given by:

$$\mathcal{L}_s = -\frac{1}{\tilde{K}}\frac{1}{L}\sum_{k=1}^{\tilde{K}}\sum_{t=1}^{L}[s_k(t)\log(\hat{s}_k(t)) + (1-s_k(t))\log(1-\hat{s}_k(t))], \tag{5.7}$$

$$\mathcal{L}_w = -\frac{1}{\tilde{K}}\sum_{k=1}^{\tilde{K}}[w_k\log(\hat{w}_k) + (1-w_k)\log(1-\hat{w}_k)], \tag{5.8}$$

where the segment index j has been omitted for simplicity of notation. The rationale behind the use of BCE loss for multi-label multi-class problems is that the task is reduced to multiple binary classification problems, one for each appliance. Individual BCE loss terms are calculated for each output neuron, then they are summed to obtain the final loss. Training has been performed by using the Adam algorithm [94].

5.1.3. Post-Processing

The bag-level and instance-level outputs of the CRNN are class probabilities estimates in the range $[0, 1]$. These values are then transformed into 0 or 1 by using a threshold to determine whether an appliance is active or not.

This procedure, however, is prone to producing outputs where few isolated instances are 0 or 1. Median filter is one of the popular solutions to reduce such spurious values. The median filter operates by calculating the median value within a segment of a certain length and replacing contiguous instances with a duration shorter than half the segment length with the median value.

Additionally to median filtering, here a recent technique presented in [95] and named *clip smoothing* is explored. Clip smoothing operates before thresholding and consists in multiplying the instance-level prediction with the bag-level prediction (Figure 5.3). The rationale of clip smoothing is that instance and bag level predictions should be coherent: if a bag prediction is close to 0, instance-level predictions should be all close to 0, and vice versa. Multiplying the two predictions enforces this relationship. An advantage over median filtering is that clip smoothing is a learnable procedure intrinsic to the network. Note that the use of clip smoothing is only possible when the network outputs weak and strong predictions, thus it represents an additional advantage over strongly supervised methods.

Table 5.1.: UK-DALE dataset characteristics. Numbers are in thousands.

	Strongly and weakly annotated set						Weakly annotated set	Average power in a activation (W)
	Training (k)		Validation (k)		Test (k)		Training (k)	
Appliances	Strong	Weak	Strong	Weak	Strong	Weak	Weak	
Kettle	996.6	31.4	196.3	6.9	91.4	2.2	11.7	1996
Microwave	849.7	31.0	157.2	7.0	83.8	2.4	11.9	1107
Fridge	1221.9	4.8	709.4	2.9	130.3	0.6	31.2	91
Washing Machine	837.7	1.2	881.4	1.2	102.5	0.2	30.9	487
Dishwasher	554.5	0.6	790.1	0.9	87.5	0.2	31.3	723
Nr. of bags	41.720		10.428		3.271		58.213	

Table 5.2.: REFIT dataset characteristics. Numbers are in thousands.

	Strongly and weakly annotated set						Weakly annotated set	Average power in a activation (W)
	Training (k)		Validation (k)		Test (k)		Training (k)	
Appliances	Strong	Weak	Strong	Weak	Strong	Weak	Weak	
Kettle	2917.3	62.2	619.2	15.5	623.9	20.9	3.0	2048
Microwave	1858	40	455.6	9.9	467.7	12.0	20.0	893
Fridge	6030	10	1635.5	3.0	1396.1	1.4	55.0	90
Washing Machine	2402.2	6.1	2062.9	5.7	228.3	0.5	55.0	513
Dishwasher	2263.2	2.9	2822.5	4.4	472.0	0.5	53.0	881
Nr. of bags	97.385		24.297		22.425		102.078	

5.2. Experimental setup

The proposed method has been implemented in Python using Tensorflow 2.4 and Keras. The source code is available here[1].

5.2.1. Dataset

The experiments have been conducted on two datasets, UK-DALE [24], and REFIT [25], already described in Section 2.3. The monitored appliances have been selected based on the recent literature [96, 97, 35], and they are the following: Kettle, Microwave, Fridge, Washing Machine, and Dishwasher. Each dataset has been processed to create two sets of bags, one for UK-DALE and one for REFIT, then used for training and testing the proposed method. The complete procedure is described in Appendix 1.

An example of aggregate segment related to house 2 of the REFIT dataset is shown in Figure 5.2. Aggregate data were normalized with mean and standard deviation values computed from the training set.

The aggregate active power readings are down-sampled from 1 s to 6 s, and the mains are aligned to the appliance readings using NILM-TK [98]. All the houses were included, but only the Kettle and the Fridge were considered for houses 3 and 4. For training and validation, data from houses 1, 3, 4, and 5 are used, while house 2 was kept out for testing on unseen data.

For REFIT, the same houses reported in [97] are used, a part from house 20 since it contains only two Kettle activations. Houses 4, 9, and 15 have been used to test on unseen data, while the remaining for training. The training,

[1] `https://github.com/GiuTan/Weak-NILM`

validation, and test set characteristics are reported in Table 5.1 and Table 5.2: "Strongly and weakly annotated set" refers to bags with both strong and weak labels, while "Weakly annotated set" refers to bags annotated only with weak labels. For each appliance, the table reports the number of strong labels, i.e., the total number of samples, and the number of weak annotations, i.e., the total number of bags where it is present.

5.2.2. Benchmark Methods

The proposed method has been compared to two benchmark approaches recently published in the literature. The first is the LSTM network presented in [28], that has been already used as benchmark method for classification in [45, 5]. As in [5], to perform multi-label classification, the last layer of the network has been replaced with a fully-connected layer composed of $K = 5$ neurons, that is the number of monitored appliances, followed by a sigmoid activation function. The network is trained only on strongly labeled data using the loss defined in Equation 5.7. The second benchmark method is the Semi-Supervised Multi-Label TCN (SSML-TCN) proposed in [5] and the network has been implemented and trained with the hyperparameters reported in the original work. The SSML-TCN network has been trained using both strongly and weakly labeled data, with the latter used as unlabeled data. The final loss function is the sum of the cross-entropy loss defined in Equation 5.7 and the consistency loss computed on the student and teacher predictions as in [5]. The network used to produce the inferences is the student one. More details about benchmark approaches' architectures and peculiarities have been reported in Appendix 2.

The proposed solution has been evaluated also against a CRNN trained only on strongly annotated data as the LSTM network. Referring to Figure 5.3, this means that this network outputs only instance-level predictions, and it does not comprise the linear softmax pooling layer and clip smoothing. This network will be denoted as S-CRNN in the following.

5.2.3. Evaluation Metrics

The performance of the algorithm has been assessed at the instance level, while the bag-level output has not been considered. The metrics used in the evaluation are the F_1-score (F_1) and the Total Energy Correctly Assigned (TECA) [26]. The F_1-score is used to evaluate the model prediction ability, balancing between the presence of accurate classification and false activations. F_1-score for appliance k is calculated as:

$$F_1^{(k)} = \frac{2 \cdot TP^{(k)}}{2 \cdot TP^{(k)} + FP^{(k)} + FN^{(k)}}, \tag{5.9}$$

where $TP^{(k)}$ is the number instances correctly assigned to appliance k (true positives), $FP^{(k)}$ is the number instances incorrectly assigned to appliance k (false positives), and $FN^{(k)}$ is the number instances incorrectly assigned to other appliances (false negatives). The average performance across appliances is calculated by using the micro-averaged F_1-score:

$$F_1\text{-micro} = \frac{2 \cdot \sum_{k=1}^{K} TP^{(k)}}{\sum_{k=1}^{K} \left(2 \cdot TP^{(k)} + FP^{(k)} + FN^{(k)}\right)}. \tag{5.10}$$

TECA has been introduced in [26] to evaluate the energy disaggregation error and is defined as:

$$\text{TECA} = 1 - \frac{\sum_k \sum_t |\hat{x}_k(t) - \bar{x}_k(t)|}{2 \sum_t \bar{y}(t)}, \tag{5.11}$$

where $\hat{x}_k(t)$ is the estimated power of appliance k at the time instant t, $\bar{x}_k(t)$ the related ground-truth power, and $\bar{y}(t) = \sum_k \bar{x}_k(t)$. The estimated power $\hat{x}_k(t)$ is reconstructed by multiplying the estimated states $\hat{s}_k(t)$ and the average power in an activation of appliance k, while $\bar{x}_k(t)$ by considering the ground-truth states $s_k(t)$. Average powers are reported in Table 5.1 and Table 5.2 respectively for the UK-DALE and REFIT datasets.

Differently from F_1-micro, TECA is more influenced by high power appliances, thus, it may result in high values even when low-power appliances are classified poorly [26].

5.2.4. Experimental procedure

Referring to the strongly and weakly annotated training sets reported in Table 5.1 and Table 5.2, three experiments are performed in different training conditions:

1. Experiment 1: all the weakly annotated training bags are used for training, while the number of strongly annotated bags is varied from 0% to 100% (step 20%);

2. Experiment 2: the amount of strongly annotated bags is fixed to 20%, while the number of weakly annotated bags is varied from 0% to 100% % (step 20%);

3. Experiment 3: mixed strongly labeled data of UK-DALE and weakly labeled data of REFIT. The objective is to evaluate if it improves the

performance on the respective test sets compared to training only on strongly labeled data.

The objective of the first two experiments is to evaluate how weakly labeled data influence performance, particularly if they indeed provide an improvement when the amount of strongly labeled data is modest.

In Experiment 1, the amount of strongly labeled data is progressively decreased, and it is evaluated when the contribution of weakly labeled data is significant. In Experiment 2, a certain amount of strongly labeled data for which weakly labeled data provide a performance improvement is considered, and then the amount of weakly labeled data is varied. In this way, the contribution of which amount of weakly labeled data provides a performance improvement can be evaluated. Note that in the first experiment, 0% of strongly annotated training data means that training is performed by using only weak supervision. Experiment 3 is the case where it is possible to acquire additional data on a target environment, but annotation is performed only with weak labels. For example, when end users perform annotation as a result of a prompt to label an aggregate power segment in which unknown loads are present. In this case, users annotate the entire segment with a weak label, thus indicating only whether an appliance was active or not. In this situation, the aim is to evaluate if mixing this additional data with strongly annotated data from a public dataset provides some benefits. To perform this evaluation, weakly labeled data and test data from REFIT have been resampled to 6 s as UK-DALE strongly labeled data.

A tuning procedure has been performed for each training condition to find the values of hyperparameters that achieve the highest performance on the validation set. The procedure has been conducted separately for the proposed method and the S-CRNN network. In this way, the possibility that the performance difference is due to a wrong or biased choice of the values of the hyperparameters is reduced.

Table 5.3.: Training hyperparameters not subject to tuning.

Parameters	Value
Batch size	64 (UK-DALE) 128 (REFIT)
Learning rate	0.002
Training epochs	1000
Patience	15
Stride	1
Padding	Same
Weights initializer	Glorot Uniform
Bias initializer	Zeros

Table 5.4.: Hyperband parameters. "Max epochs" refers to epochs for the Hyperband algorithm thus the number differs from the epochs of the learning process. U is the number of GRUs, H is the number of convolutional blocks, K_e is the kernel dimension and p is the dropout probability.

Parameters	[Range], Step	Distribution
Max epochs	20	-
Factor	2	-
U	[8, 16, 32, 64, 128, 256]	Random choice
H	[2, 6], 1	Uniform
K_e	[3, 7], 2	Uniform
p	[0.1, 0.5], 0.1	Uniform

Hyperband [99] has been adopted to improve the search of CRNN hyperparameters. This is a method that uses random sampling and early-stopping principles. It begins by randomly selecting many hyperparameter settings and giving each a small amount of resources. It then gradually removes the configurations that perform the worst, redistributing their resources to the remaining configurations. This elimination process is done in stages until only the best configurations are left. Hyperband has been used for searching the following hyperparameters: number of convolutional layers (H), number of units in the recurrent layers (U), the dropout rate (p), and kernel size (K_e). The number of filters F in each convolutional layer increases doubling layer by layer with an initial value of 32. Table 5.3 reports the values of the hyperparameters not subject to tuning with Hyperband, Table 5.4 the hyperparameters of Hyperband, and Table 5.5 the values determined after tuning for the different training conditions. The value of the weight λ, that balances the strong and weak loss contributions, has been initially set to 1. Then, monitoring the values assumed by the two losses $\mathcal{L}_s$ and $\mathcal{L}_w$, the final value of λ is selected to make them of the same order of magnitude.

5.2.5. Post-processing

Whether to apply median filtering, clip smoothing, or none of the two is selected by evaluating the results obtained on the validation set. Median filtering did not improve the classification performance, so it was not used.

The threshold for obtaining the final classification values from output probabilities has been selected on the validation set, based on the value that maximizes the F_1-score.

Table 5.5.: Hyperparameters determined after tuning.

Dataset	% Weak	% Strong	H	U	K_e	p
UKDALE	100	0	4	16	5	0.1
		20-100	3	64	5	0.1
	0	20-100	3	64	5	0.1
	20-100	20	3	64	5	0.1
REFIT	0	20	4	256	5	0.3
		40	3	64	3	0.2
		60	3	128	3	0.2
		80	4	128	7	0.3
		100	5	64	3	0.2
	20, 40		4	256	5	0.3
	60, 80	20	3	64	3	0.1
	100		3	64	3	0.3
		40	4	64	5	0.1
	100	60	4	64	3	0.2
		80, 100	4	64	5	0.1
		0	3	32	3	0.1

Table 5.6.: Maximum model size, training and test time of all the evaluated methods.

Method	Max Model Size	Training Time	Testing Time
LSTM	4.97 MB	172 ms/step	3.6 ms
SSML-TCN	6.18 MB	143 ms/step	4 ms
S-CRNN	4 MB	215 ms/step	0.3 ms
Proposed	1.39 MB	214 ms/step	0.3 ms

5.2.6. Complexity Details

As a first note, Table 5.6 reports the maximum model size, and the training and inference times for all the evaluated methods. Note that the network of the proposed approach is the smallest of the evaluated methods and, along with S-CRNN, requires the least amount of time for testing. On the other hand, it requires more time for training, as S-CRNN, compared to other methods. Training and test times have been obtained on a NVIDIA DGX Station A100 [100]. For training time and model size, there have been reported the maximum values related to the longest training, both for UK-DALE and REFIT.

Table 5.7.: Results obtained on the UK-DALE and REFIT datasets by using weakly labeled data only, in terms of F_1-score (Section 5.3).

	0% Strong, 100% Weak						
	KE	MW	FR	WM	DW	F_1-micro	TECA
UKDALE	0.89	0.76	0.29	0.36	0.39	0.52	0.57
REFIT	0.21	0.17	0.01	0.09	0.17	0.11	0.06

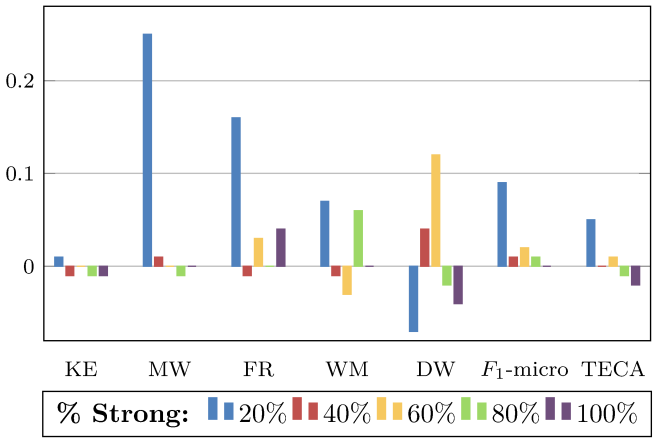


Figure 5.4.: Difference between F_1-scores of each appliance, F_1-micro, and TECA of the proposed method and S-CRNN for UK-DALE for the different percentages of strongly labeled data (Section 5.3).

5.3. Results experiment 1: Fixed amount of weakly labeled data

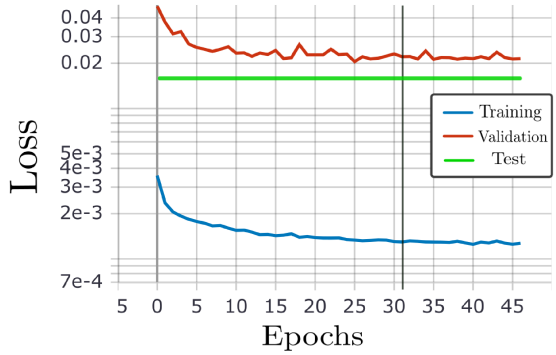
5.3.1. UK-DALE

The results related to this experiment are reported in Table 5.7, Table 5.8, and Figure 5.4. Table 5.7 shows the results obtained by using only weak labels for training. Observing the results, Kettle and Microwave F_1-scores are above 0.75, with the former equal to 0.89. On the contrary, Fridge, Washing Machine, and Dishwasher scores are below 0.5, meaning that the absence of strong labels impacts their results more than other appliances.

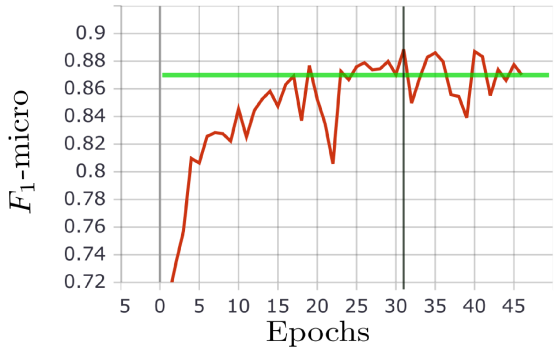
Table 5.8 reports the results obtained when strongly labeled data are used concurrently with weak labels. In terms of F_1-micro, apart when 100% of strongly labeled data is used, the proposed method provides better performance with respect to benchmark approaches. In terms of TECA, the S-CRNN achieves the overall greatest value, but on average the proposed method achieves superior performance. In particular with 20%, 40% and 60% of strongly

Table 5.8.: Results obtained on the UK-DALE dataset related to Experiment 1. Best scores for each strong percentage are highlighted in bold. Best score among all the percentage are underlined (Section 5.3).

% Strong	Method	KE	MW	FR	WM	DW	F_1-micro	TECA
20	LSTM [28]	0.95	0.74	0.35	0.44	0.69	0.61	0.79
	SSML-TCN [5]	0.82	0.70	0.16	0.39	0.60	0.46	0.60
	S-CRNN	0.98	0.67	0.42	0.80	**0.81**	0.72	0.86
	Proposed	**0.99**	**0.92**	**0.58**	**0.87**	0.74	**0.81**	**0.91**
40	LSTM [28]	**0.99**	0.93	0.59	0.59	**0.88**	0.77	0.89
	SSML-TCN [5]	0.92	0.86	0.37	0.62	0.48	0.64	0.77
	S-CRNN	**0.99**	0.95	**0.70**	**0.88**	0.84	0.86	**0.94**
	Proposed	0.98	**0.96**	0.69	0.87	**0.88**	**0.87**	**0.94**
60	LSTM [28]	**0.99**	0.93	0.53	0.69	**0.84**	0.76	0.89
	SSML-TCN [5]	0.95	0.87	0.39	0.70	0.64	0.68	0.82
	S-CRNN	**0.99**	**0.96**	0.67	**0.90**	0.71	0.84	0.93
	Proposed	**0.99**	**0.96**	**0.70**	0.87	0.83	**0.86**	**0.94**
80	LSTM [28]	**0.99**	0.95	0.58	0.68	0.69	0.75	0.88
	SSML-TCN [5]	0.96	0.84	0.41	0.76	0.60	0.68	0.83
	S-CRNN	**0.99**	**0.96**	**0.70**	0.83	**0.89**	0.86	**0.94**
	Proposed	0.98	0.95	**0.70**	**0.89**	0.87	**0.87**	0.93
100	LSTM [28]	**0.99**	0.95	0.65	0.78	0.75	0.80	0.91
	SSML-TCN [5]	0.97	0.85	0.43	0.76	0.61	0.71	0.84
	S-CRNN	**0.99**	**0.96**	0.70	**0.89**	**0.91**	**0.88**	**0.95**
	Proposed	0.98	**0.96**	**0.74**	**0.89**	0.86	**0.88**	0.93
AVG.	LSTM [28]	0.98	0.90	0.54	0.64	0.77	0.74	0.87
	SSML-TCN [5]	0.92	0.82	0.35	0.65	0.59	0.63	0.77
	S-CRNN	**0.99**	0.90	0.64	0.86	0.83	0.83	0.92
	Proposed	0.98	**0.95**	**0.68**	**0.88**	**0.84**	**0.86**	**0.93**



(a) Training, validation, and test losses.



(b) Validation and test F_1-micro.

Figure 5.5.: Training loss and validation loss and F_1-score for the experiment related to 40% strong data and 100% weak data for UK-DALE. Vertical bar indicates the early stopping epoch.

labeled data, i.e., when the number of strong labels is modest, the proposed method shows more accuracy. Considering the average across the different percentages of strongly labeled data (last line of Table 5.8), the proposed method significantly improves the performance of all the appliances, with the only exception of Kettle. On average, the F_1-micro improvements compared to LSTM, SSML-TCN, and S-CRNN are respectively 16.22%, 36.51%, and 3.61%. Among benchmark methods, S-CRNN performs more accurately compared to LSTM and SSML-TCN.

Figure 5.4 shows the difference between the F_1-scores of each appliance, the F_1-micro, and the TECA of the proposed method and S-CRNN for the different percentages of strongly labeled data. S-CRNN has been chosen among benchmark methods since it is the best performing among them. Moreover, it allows highlighting the contribution of weak labels since the architecture is very similar to the one of the proposed method. It is evident that the greatest improvement occurs when the percentage of strongly labeled data is 20%, i.e.,

when the difference between the amount of strongly and weakly labeled data is the largest, meaning that in this case weak labels influence more the learning phase. Apart from the Dishwasher, the improvement is consistent for all the appliances.

Above 20%, the improvement of the proposed method reduces, but it remains significant up to 100%. In this case, the F_1-micros are comparable, meaning that the contribution of weak labels is less important. Observing the performance of the individual appliances, weak labels influence to a lesser extent the performance of Kettle and Microwave.

The appliances that exhibit a less consistent behavior with weak labels are Dishwasher and Washing Machine. Regarding the former, with 40% and 60% of strongly labeled data, the proposed method improves the performance with respect to full supervision, while with 20%, 80%, and 100% the performance is lower. The same holds for Washing Machine where the performance improves with 20% and 80% of strongly labeled data, while in the other cases weak supervision does not improve the classification ability. A possible explanation for this behavior can be related to the shape of the activations of these appliances, which are more complex compared to the others, as also reported in the previous literature [97].

Observing the results of the individual appliances, for Kettle and Microwave weak labels allow to use a less amount of strong labels for obtaining the same F_1-score. For the Fridge, on the other hand, weak labels provide the overall best performance when 100% of strongly labeled data is used.

Figure 5.5 shows an example of the loss trend for training, validation, and test, the F_1-micro trend during training, and the final value on the test set computed by using the best model. Early stopping occurs on the 46th epoch.

5.3.2. REFIT

REFIT is a more challenging dataset than UK-DALE as it is significantly noisier [101]. Indeed, the results shown in Table 5.7 obtained by using only weakly labeled data are lower compared to the ones obtained with UK-DALE leading to the conclusion that weakly labeled data only are not sufficient to achieve satisfactory performance.

Table 5.9 reports the results with a fixed amount of weakly labeled data and varying percentages of strongly labeled data. Observing the F_1-micro for the different percentages and the average value, the proposed method achieves superior performance compared to benchmark methods, with the only exception of 60% of strongly labeled data where S-CRNN performs the same. The best F_1-micro is reached with the proposed method when the percentage of strongly labeled data is 40%. In terms of TECA, on average, S-CRNN and the proposed

Table 5.9.: Results obtained on the REFIT dataset related to Experiment 1. Best scores for each strong percentage are highlighted in bold. Best score among all the percentage are underlined (Section 5.3).

% Strong	Method	KE	MW	FR	WM	DW	F_1-micro	TECA
20	LSTM [28]	**0.86**	0.53	0.23	0.46	0.67	0.51	**0.74**
	SSML-TCN [5]	0.72	0.71	0.12	0.59	0.51	0.42	0.58
	S-CRNN	0.68	0.40	0.29	**0.68**	**0.68**	0.44	0.65
	Proposed	0.68	**0.80**	**0.50**	0.54	0.58	**0.59**	0.65
40	LSTM [28]	0.84	0.77	0.25	0.27	0.70	0.54	0.76
	SSML-TCN [5]	0.81	0.71	0.08	0.52	0.47	0.39	0.65
	S-CRNN	**0.85**	**0.83**	0.28	**0.72**	0.76	0.62	**0.81**
	Proposed	0.74	0.80	**0.45**	0.59	**0.85**	**0.63**	0.76
60	LSTM [28]	0.67	0.80	0.31	0.48	0.46	0.51	0.71
	SSML-TCN [5]	0.74	0.70	0.10	0.61	0.48	0.36	0.63
	S-CRNN	**0.81**	**0.86**	0.35	**0.72**	0.63	**0.62**	**0.79**
	Proposed	0.78	0.82	**0.40**	0.54	**0.82**	**0.62**	0.77
80	LSTM [28]	0.70	0.77	**0.43**	0.59	0.69	0.58	0.73
	SSML-TCN [5]	**0.80**	0.74	0.08	0.63	0.55	0.43	0.70
	S-CRNN	0.76	0.75	0.32	**0.76**	0.81	0.60	**0.77**
	Proposed	0.58	**0.79**	0.41	**0.76**	**0.89**	**0.61**	0.74
100	LSTM [28]	0.57	0.80	**0.43**	0.53	0.31	0.51	0.67
	SSML-TCN [5]	**0.77**	0.73	0.11	0.60	0.52	0.42	0.68
	S-CRNN	0.64	0.80	0.33	0.65	0.65	0.55	0.71
	Proposed	0.73	**0.84**	0.36	**0.77**	**0.78**	**0.62**	**0.78**
AVG.	LSTM [28]	0.73	0.73	0.33	0.47	0.57	0.53	0.72
	SSML-TCN [5]	**0.77**	0.72	0.10	0.59	0.51	0.40	0.65
	S-CRNN	0.75	0.73	0.31	**0.71**	0.71	0.57	**0.75**
	Proposed	0.70	**0.81**	**0.42**	0.64	**0.78**	**0.61**	0.74

method achieve similar results, with the former obtaining a value 0.01 greater. The performance of the appliances with the highest average power consumption in an activation and the composition of the test set influence the behavior for the different percentages of strongly labeled data. As shown in Table 5.2, the Kettle is the appliance with the highest power consumption, and with 20%, 40%, and 60% of strongly labeled data, the method with the greatest F_1-score on the Kettle also achieves the highest TECA. When the percentage of strongly labeled data is 80% and 100%, SSML-TCN achieves the highest F_1-score, but the overall F_1-micro is significantly lower than the proposed method and S-CRNN, and the value of TECA is consequently lower. However, it is worth remarking that the proposed method achieves an average TECA close to the one of the S-CRNN, while providing a higher F_1-micro..

Regarding individual appliances, in terms of average F_1-scores, the proposed method achieves the greatest performance for Microwave, Fridge, and Dishwasher, while SSML-TCN for Kettle and S-CRNN for Washing Machine. The best F_1-scores across all the percentages (underlined results in Table 5.9) are obtained by using the proposed method for Washing Machine, Dishwasher, and

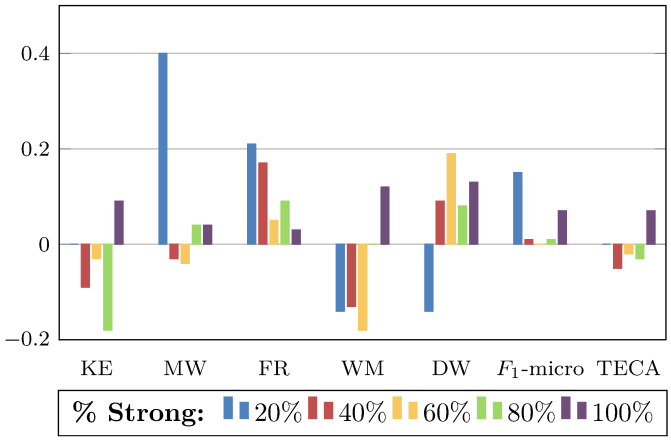


Figure 5.6.: Difference between F_1-scores of each appliance, F_1-micro, and TECA of the proposed method and S-CRNN for REFIT for the different percentages of strongly labeled data (Section 5.3).

Fridge, while with LSTM for the Kettle and S-CRNN for the Microwave.

Figure 5.6 shows the difference between the F_1-scores of each appliance, the F_1-micro, and the TECA of the proposed method and S-CRNN. The focus is on S-CRNN as with UK-DALE for the same reasons, i.e., since it is the best performing among benchmark methods, and it allows to highlight the contributions of weak labels. Microwave, Fridge and Dishwasher are the appliances that benefit most from weak labels during training, in particular when strong data are only 20%. On the other hand, Kettle and Washing Machine exhibit the greatest benefit from weak labels when the amount of strongly labeled data is large.

5.4. Results experiment 2: Fixed amount of strongly labeled data

As aforementioned, in this experiment the amount of strongly labeled data is fixed and the amount of weakly labeled data varies. Both for UK-DALE and REFIT, the percentage of strongly labeled data is fixed to 20%, the lowest value considered in Experiment 1. In this experiment, the objective is to evaluate to what extent weakly labeled data influence the performance when the amount of strongly labeled data is modest. For each percentage, only the proposed method and SSML-TCN are trained. The other benchmark methods do not use weakly labeled data for training thus they are excluded from this experiment. For the sake of conciseness, in Table 5.10 and Table 5.11, only the results of

Table 5.10.: Results obtained on the UK-DALE dataset related to Experiment 2. The best results obtained using the least amount of weakly labeled data are highlighted in bold. (Section 5.4)

% Weak	Method	KE	MW	FR	WM	DW	F_1-micro	TECA
0	S-CRNN	0.98	0.67	0.42	0.80	0.81	0.72	0.86
20	SSML-TCN	0.85	0.71	0.19	0.46	0.65	0.54	0.68
	Proposed	0.98	**0.93**	**0.58**	0.79	**0.84**	**0.81**	**0.91**
40	SSML-TCN	0.94	0.64	0.21	0.54	0.66	0.56	0.76
	Proposed	**0.99**	0.92	0.51	0.81	0.69	0.77	0.89
60	SSML-TCN	0.89	0.66	0.21	0.51	0.68	0.56	0.73
	Proposed	0.99	0.92	0.58	0.82	0.82	0.81	0.91
80	SSML-TCN	0.83	0.73	0.18	0.39	0.63	0.50	0.63
	Proposed	0.98	0.92	0.53	0.83	0.81	0.80	0.91
100	SSML-TCN	0.82	0.70	0.16	0.39	0.60	0.46	0.60
	Proposed	0.99	0.92	0.58	**0.87**	0.74	0.81	0.91
AVG.	SSML-TCN	0.87	0.69	0.19	0.46	0.64	0.52	0.68
	Proposed	**0.99**	**0.92**	**0.56**	**0.82**	**0.78**	**0.80**	**0.91**

the proposed method, SSML-TCN, and S-CRNN are reported since it is the method that achieved the best average performance in Experiment 1.

5.4.1. UK-DALE

Table 5.10 presents the results related to the UK-DALE dataset. Observing the results, in terms of F_1-micro, introducing 20% of weak labels allows achieving the highest performance. Indeed, introducing more weak data does not provide significant improvements in that sense. In terms of TECA, the greatest value is obtained by using 20%, 60%, 80%, and 100% of weakly labeled data. Compared to S-CRNN and SSML-TCN, the proposed method always achieves greater F_1-micro and TECA.

Regarding individual appliances, the greatest average F_1-micro is always achieved by using the proposed method. The highest F_1-scores for most appliances are obtained with the lower percentages of weak data (20% and 40%). The F_1-score of Kettle and Microwave is almost independent of the number of weak labels since it changes only by 0.01. Instead, the F_1-score of the Washing Machine improves constantly with the increase of weakly labeled data used. The Dishwasher exhibits a significant improvement by using 20% of weak data, then the behavior is less consistent. A possible explanation is that the performance is more influenced by the composition of the weak dataset and the related unbalance of the classes.

In fact, the Dishwasher is significantly unbalanced considering weak annotations with a presence of 0.89%, with respect to the total presences of all the appliances in the dataset when weakly annotated data considered are 40%. In

Table 5.11.: Results obtained on the REFIT dataset related to Experiment 2. The best results obtained using the least amount of weakly labeled data are highlighted in bold (Section 5.4).

% Weak	Method	KE	MW	FR	WM	DW	F_1-micro	TECA
0	S-CRNN	0.68	0.40	0.29	0.68	0.68	0.44	0.65
20	SSML-TCN	**0.74**	0.74	0.06	0.54	0.32	0.36	0.54
	Proposed	0.72	0.70	0.38	**0.77**	0.69	0.58	**0.73**
40	SSML-TCN	0.73	0.72	0.10	0.60	0.30	0.37	0.49
	Proposed	0.66	**0.85**	0.36	0.77	0.66	**0.59**	0.73
60	SSML-TCN	0.72	0.69	0.09	0.53	0.30	0.34	0.49
	Proposed	0.67	0.74	0.28	0.68	**0.74**	0.54	0.70
80	SSML-TCN	0.72	0.71	0.07	0.49	0.45	0.37	0.58
	Proposed	0.60	0.81	0.39	0.69	0.70	0.58	0.68
100	SSML-TCN	0.72	0.71	0.12	0.59	0.51	0.42	0.58
	Proposed	0.68	0.80	**0.50**	0.54	0.58	0.59	0.65
AVG.	SSML-TCN	**0.73**	0.71	0.09	0.55	0.38	0.37	0.54
	Proposed	0.67	**0.78**	**0.38**	**0.69**	**0.67**	**0.58**	**0.70**

fact, for 20% the presence is about 1.4%, for 60% is 3.4%, for 80% is 9.6% and for 100% is 16.9%.

5.4.2. REFIT

Table 5.11 reports the results related to the REFIT dataset. Generally, the F_1-micro related to the proposed method for different percentages of weakly labeled data does not change significantly, apart for 60%. Regardless the percentage, the proposed method always outperforms SSML-TCN and S-CRNN in terms of F_1-micro and the highest value is obtained for 40% of weakly labeled data. In terms of TECA, the proposed method outperforms both S-CRNN and SSML-TCN, achieving the overall greatest value with 20% and 40% of weakly labeled data.

Regarding individual appliances, on average, the highest F_1-scores are achieved by using the proposed method with the only exception of the Kettle. For the different weakly labeled data percentages, the F_1-scores behaves differently depending on the appliance, but generally highest scores occur for lower percentages (20%-40%). This applies to the Kettle, Washing Machine and Microwave, while for the Dishwasher the best F_1-score is obtained when the percentage is 60% and for the Fridge when it is 100%. For the Microwave, the F_1-score is always higher than the one of the S-CRNN method. SSML-TCN achieves the highest F_1-score for the Kettle. However, the proposed method classifies the Kettle better than the S-CRNN when the weak data are modest (20%).

Table 5.12.: Results obtained on the UK-DALE test set with mixed training set. Best scores are reported in bold (Section 5.5).

	KE	MW	FR	WM	DW	F_1-micro	TECA
S-CRNN	**0.98**	0.67	**0.42**	**0.80**	0.81	0.72	0.86
Proposed (Mixed)	0.96	**0.75**	0.34	0.79	**0.88**	**0.75**	**0.88**

Table 5.13.: Results obtained on REFIT test set with mixed training set. Best scores are reported in bold (Section 5.5).

	KE	MW	FR	WM	DW	F_1-micro	TECA
S-CRNN	0.68	0.40	**0.29**	**0.68**	0.68	0.44	0.65
Proposed (Mixed)	**0.78**	**0.45**	0.21	0.43	**0.74**	**0.47**	**0.68**

5.5. Results experiment 3: Mixed training set

In this experiment it is evaluated whether mixing weakly labeled data of REFIT and strongly labeled data of UK-DALE during training improves the performance compared to S-CRNN on the test sets of both datasets. Among the benchmark approaches, S-CRNN is chosen since it is the best performing, and it allows us to highlight the contribution of weakly labeled data since its architecture is similar to that of the proposed method. The percentage of UK-DALE strongly labeled training set is 20%.

As shown in Table 5.12 for the UK-DALE dataset, the proposed network trained on mixed datasets improves both F_1-micro and TECA with respect to supervised learning. In particular, for Microwave and Dishwasher, the improvement is consistent, while for Kettle, Fridge, and Washing Machine, the performance slightly deteriorates.

On the REFIT test set, F_1-micro improves by 6.8% when the mixed training set is used compared to when training is performed only on strongly labeled REFIT data (Table 5.13). TECA is also higher for the proposed method, with a 4.6% improvement over S-CRNN. Note, however, that the F_1-score of all appliances increases, and the only exceptions are the Fridge and the Washing Machine. This result is coherent to what was reported in [97], where Washing Machine was the only appliance with lower performance when training and testing were performed on different datasets. Moreover, consider also that in proposed case, only the UK-DALE validation set is used (Table 5.1) for hyperparameters optimization, and Washing Machine is the appliance having the largest quantity of strong labels compared to the others. Nonetheless, this result evidences how a modest quantity of strong data with weak annotations can positively enhance classification on unseen data for most appliances.

Chapter 6.

A Multiple Instance Regression Approach to Electrical Load Disaggregation

The encouraging outcomes of the previous method have led to an expanded strategy for reconstructing the power consumption profile of appliances. This expansion benefits the user by reducing the costs associated with sensor installation, as mentioned in Chapter 1.

Given the unique nature of the problem, it is worth to discuss Multiple Instance Regression (MIR) [102]. To the best of the author's knowledge, weak supervision has not been previously used to estimate an appliance's active power.

This chapter introduces a MIR-based approach for electrical load disaggregation. In MIR, the goal is to predict multiple real-valued variables using weak labels for training. In this context, real-valued variables are samples of an appliance's active power, representing strong labels. Conversely, a weak label is the average value of a segment of the appliance's active power values.

The method utilizes both weak and strong labels to train a CRNN for load disaggregation. Therefore, NILM is modeled as a MIR problem, and the availability of weakly labeled data is leveraged to decrease the amount of strongly labeled data needed in supervised approaches, thereby enhancing performance.

This work has been presented and published in the proceedings of the European Signal Processing Conference (EUSIPCO) in 2022 [103].

6.1. Proposed Methodology

Instances are represented by samples of power reading of the mains $y(t)$. $y(t)$ is divided into windows of fixed length L and overlapped by $P < L$ samples. The bag j is defined as the j-th window of $y(t)$ as follows:

$$\mathbf{y}_j = [y(j(L-P)), \ldots, y(j(L-P)+L-1)]^T. \tag{6.1}$$

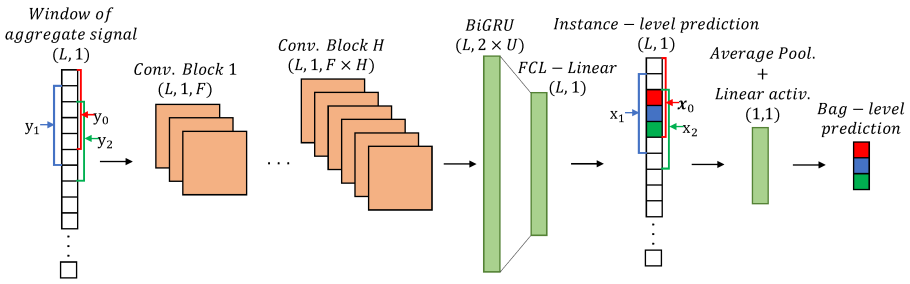


Figure 6.1.: The proposed architecture for MIR-based NILM.

Omitting the device index k for simplicity of notation, the strong labels for bag $\mathbf{y}_j$ of a generic appliance are represented by the ground-truth data $\mathbf{x}_j = [x(j(L-P)), \ldots, x(j(L-P)+L-1)]^T$.

The weak label of a bag depends on the strong labels of the instances within the bag itself and, as explained above, the relationship is modeled by a pooling function. Several alternatives have been proposed in the literature for regression [104]. In this work, the weak label w_j related to bag $\mathbf{y}_j$ and a generic appliance is a scalar quantity calculated as the arithmetic average of the instance labels:

$$w_j = \frac{1}{L}\sum_{l=0}^{L-1} x_n(j(L-P)+l). \tag{6.2}$$

With the above definitions, it is possible to formulate load disaggregation using MIR more formally. Denoting with

$$\mathcal{T} = \{(\mathbf{y}_1, w_1, \mathbf{x}_1), \ldots, (\mathbf{y}_M, w_M, \mathbf{x}_M), (\mathbf{y}_{M+1}, w_{M+1}), \ldots, (\mathbf{y}_{M+B}, w_{M+B})\}, \tag{6.3}$$

a set of $M+B$ bags, in which M are annotated with strong and weak labels and B only with weak labels, the goal is to learn a mapping function $\mathbf{f} : \mathbb{R}^L \to \mathbb{R}^L$ from $\mathcal{T}$ for estimating the active power $\mathbf{x}$ of an appliance given a bag of unknown aggregate power $\mathbf{y}$. The mapping function $\mathbf{f}(\cdot)$ is represented by a CRNN.

6.2. Neural Network and Learning

Load disaggregation is addressed by using the CRNN described in Section 5.1.1. Specifically, here a different CRNN is trained for each appliance of interest.

The network takes a bag $\mathbf{y}$ of aggregate power as input and has two outputs: a strong-level output and a weak-level output. The first provides an estimate of the active power $\hat{\mathbf{x}}$. Supposing that L is odd, and P is even, since the aggregate signal is processed in partially overlapped windows and $\mathbf{y}$ and $\hat{\mathbf{x}}$ are of the

same length L, only the $L - P$ central value of the window output is retained. The entire output sequence is reconstructed by joining the individual output segments. The weak-level output is represented by the average of the instance-level predictions, consistently with equation Equation 6.2. The difference with the CRNN architecture described in Section 5.1.1 is the last layer that is a Fully-Connected Layer with a linear activation function for generating the strong-level predictions associated with the input window. The strong-level output is further processed by an average pooling layer, followed by a linear activation function in order to generate the weak-level prediction. The CRNN architecture and the sliding window approach are depicted in Figure 6.1.

Given a set of annotated bags $\mathcal{T}$, the network is trained by using a loss $\mathcal{L} = \mathcal{L}_s + \lambda\mathcal{L}_w$ given by the weighted sum of the loss associated with the strong-level output $\mathcal{L}_s$, and the one related to the weak-level output $\mathcal{L}_w$. The term λ is a weight that balances the contribution of the two losses. Both $\mathcal{L}_s$ and $\mathcal{L}_w$ are calculated as the Mean Squared Error between the related prediction and the target.

Considering a mini-batch containing J bags and a generic appliance, the two losses are calculated as follows:

$$\mathcal{L}_s = \frac{1}{J \cdot L} \sum_{j=0}^{J-1} \sum_{l=0}^{L-1} \left[x(j(L-P)+l) - \hat{x}(j(L-P)+l)\right]^2, \tag{6.4}$$

$$\mathcal{L}_w = \frac{1}{J} \sum_{j=0}^{J-1} \left(w_j - \hat{w}_j\right)^2. \tag{6.5}$$

6.3. Experiments

This section describes the experiments conducted to evaluate the proposed method.

The experiments have been carried out by using the UK-DALE dataset [24]. The appliances considered in the experiments are Microwave (MW), Fridge (FR), Dishwasher (DW), Washing Machine (WM), and Kettle (KE). All houses were included but, for houses 3 and 4, only Kettle and Fridge were considered. The periods considered for each house are 2013/04/12-2015/01/05 for house 1, 2013/05/22-2013/10/10 for house 2, 2013/02/27-2013/04/08 for house 3, 2013/03/09-2013/10/01 for house 4, and 2014/06/29-2014/11/13 for house 5. The aggregate active power is down-sampled to 6 s and aligned to the appliance readings using NILM-TK [105]. Weak ground-truth labels have been created by using equation Equation 6.2. The experiments have been conducted on an *unseen* scenario, using house 2 data only for testing and data of the other houses to train and validate the model. For each appliance, the original dataset

is divided into training (1,200,000 samples), validation (150,000 samples), and testing sets (2,100,000 for Kettle and 1,700,000 samples for the other appliances).

6.3.1. Experimental procedure

Evaluation has been conducted in multiple training conditions, each characterized by a different amount of strong labels. Two extreme situations are considered: one where the amount of strong labels is very scarce, i.e., 5% of the total number of strongly annotated bags in the training set, and one where it is large, i.e., 100% the total amount in the training set. Moreover, three intermediate values such as 20%, 40%, and 80% are considered, thus each time doubling the amount of strong labels for training. The number of weak labels, on the other hand, is always the same. For each appliance and training condition, the aggregate signal is standardized by using mean and standard deviation calculated from the training set and min-max normalization to the target values is applied.

For each training condition, the approach is compared to the performance of the Sequence-to-Point approach proposed in [29] implemented with a Convolutional Network. More details on this benchmark approach are reported in the Appendix Section 2.3. Also, the same CRNN network depicted in Figure 6.1 is used as benchmark but without the bag-level output. Thus, training has been performed only on strong labels in these cases.

Both for the proposed and the comparative methods, a different network is trained for each appliance of interest by setting the maximum number of *epochs* to 1000 and using the Early-Stopping regularization technique with *patience* equal to 15 epochs. During the learning process,the ADAM optimizer [94] is used, with a learning rate equal to 0.001, β_1 and β_2 equal to 0.9 and 0.999, respectively, and ϵ equal to 10^{-7}.

6.3.2. Hyperparameters

The loss weight λ has been set to 1. Training has been performed in mini-batches, where the batch size has been determined on the validation set to minimize the error. The length L of the window is different for each appliance, and it has been determined by evaluating the performance on the validation set of the values reported in [29]. The determined values are 289 samples (29 minutes) for Microwave, 1025 samples (1 hour and 42 minutes) for Washing Machine, and 599 (1 hour) samples for Fridge, Dishwasher, and Kettle. Differently, the amount of overlap P is equal for all the appliances and has been set to $(L-1)$. This means that for each input bag, only the central value of the output is retained.

6.3.3. Evaluation metrics

The metrics used to evaluate the performance of the method are the Mean Absolute Error (MAE) and the Normalized Error in assigned Power (NEP) [105]. Both metrics are calculated for each appliance.

MAE and NEP are defined as:

$$\mathrm{MAE} = \frac{1}{T}\sum_{t=1}^{T}\left|x(t) - \hat{x}(t)\right|, \quad \mathrm{NEP} = T \cdot \frac{\mathrm{MAE}}{\sum_{t=1}^{T} x(t)}, \tag{6.6}$$

where $\hat{x}(t)$ is the power predicted by the network, $x(t)$ is the corresponding ground-truth value, and T is the number of samples of the segment under evaluation. Basically, NEP is the MAE normalized by the total appliance's energy consumption, and it allows to evaluate the importance of the error based on the appliance operating characteristics.

6.4. Results

The results obtained for each appliance and training condition are reported in Table 6.1. The proposed method is indicated with Proposed, the CRNN trained only with strong labels with CRNN-Strong, while the Sequence-to-Point network with Seq2Point.

The obtained results show that regardless of the percentage of strongly labeled data used for training, the proposed method based on weak labels is able to outperform the comparative algorithms. Compared to CRNN-Strong, MAE reduces by 3.06 W on average, while compared to Seq2Point by 8.88 W. Similarly, NEP reduces by 9.97 percentage points (pp) compared to CRNN-Strong and by 30.59 pp compared to Seq2Point.

Observing the performance for the different percentages of strong labels, the most remarkable improvement occurs when the percentage of strongly labeled data is low, i.e., 5%, 20%, and 40%, both when the proposed method is compared to CRNN-Strong and when it is compared to Seq2Point. This behavior confirms that weak labels contribute the most to improving the performance when the amount of strongly labeled data is scarce compared to weakly labeled data. Another remarkable advantage of the proposed method is the reduction of strongly labeled data quantity required to obtain the lowest error. As it can be seen for Microwave, Fridge and Dishwasher, the lowest error among all the percentages is achieved with weakly labeled data and when the quantity of strongly labeled data is only 20% while for Washing Machine only the 5%.

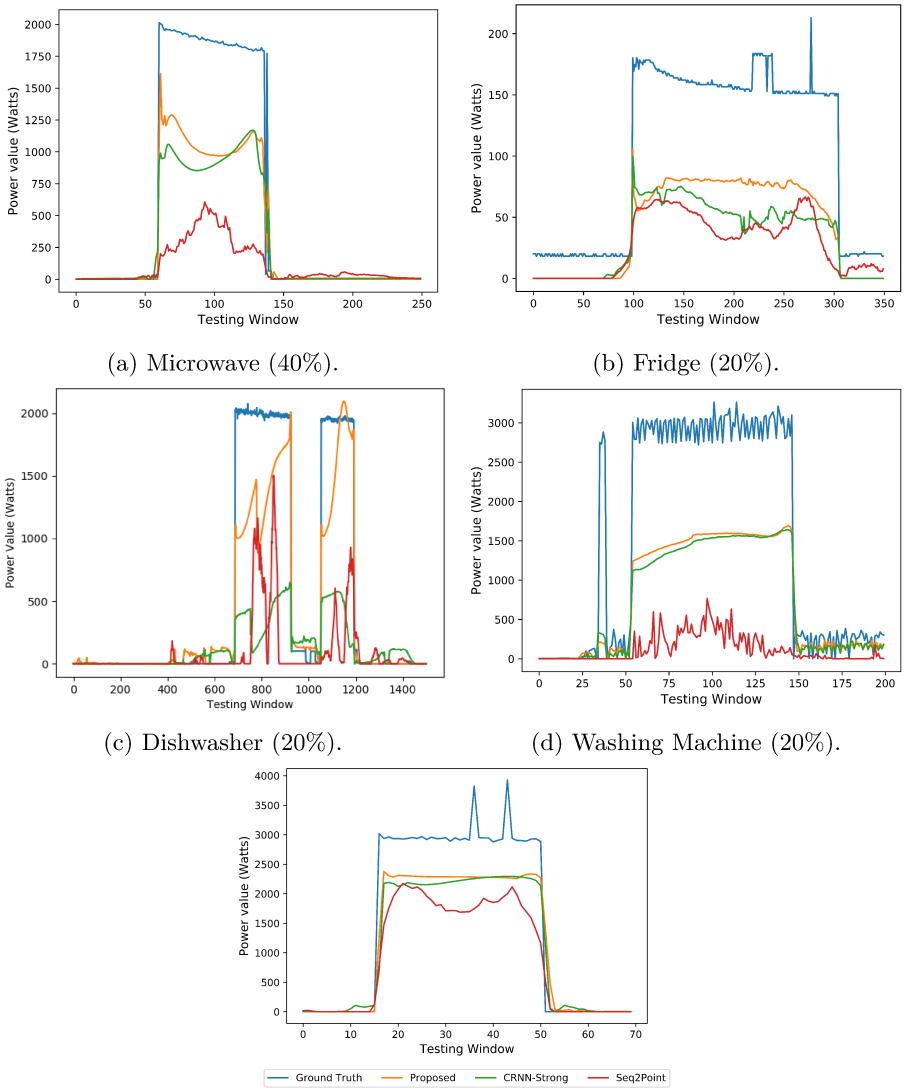
A closer look at the behavior for the different appliances shows that in the majority of the cases, the performance of the proposed method is superior to the comparative methods. The only exceptions are Microwave when the

percentage of strong labels is 80%, and Fridge and Washing Machine when the percentage is 100%. Note, however, that the MAE difference is below 0.5 W and that this occurs when the amount of strongly and weakly labeled data is comparable: in this case, the influence of weak labels is less, a behavior that could have been expected.

Figure 6.2 shows the ground-truth and the estimated active power for the proposed and comparative methods when training is performed with different percentages of strongly labeled data. The plots confirm the obtained results, as the active power outputs produced using the proposed method are closer to the ground-truth.

Table 6.1.: Results obtained for the different training conditions and addressed methods (Section 6.4). Best results for each appliance and percentage of strong labels are reported in bold.

% Strong	Model	Metric	Appliance					
			MW	**FR**	**DW**	**WM**	**KE**	**Average**
5%	Proposed	MAE (W)	**11.17**	**49.89**	**20.55**	**10.93**	**12.57**	**21.02**
		NEP (%)	**113.82**	**59.69**	**48.98**	**70.56**	**42.52**	**67.11**
	CRNN-Strong	MAE (W)	11.38	50.04	32.13	12.05	13.46	23.81
		NEP (%)	115.95	59.86	76.57	77.78	45.53	75.14
	Seq2Point	MAE (W)	15.31	66.07	48.96	14.45	28.72	34.70
		NEP (%)	155.92	79.05	116.69	93.23	97.14	108.41
20%	Proposed	MAE (W)	**8.16**	**46.93**	**20.08**	**11.44**	**12.37**	**19.80**
		NEP (%)	**83.10**	**56.15**	**47.86**	**73.82**	**41.84**	**60.55**
	CRNN-Strong	MAE (W)	8.32	47.16	37.28	12.65	14.96	24.07
		NEP (%)	84.75	56.42	88.85	81.65	50.59	72.45
	Seq2Point	MAE (W)	11.15	66.15	31.57	12.99	29.01	30.17
		NEP (%)	113.54	79.14	75.24	83.83	98.12	89.97
40%	Proposed	MAE (W)	**10.61**	**48.07**	**28.12**	**11.70**	**11.52**	**22.00**
		NEP (%)	**108.09**	**57.52**	**67.02**	**75.48**	**38.97**	**69.42**
	CRNN-Strong	MAE (W)	15.29	53.79	33.33	12.49	12.63	25.51
		NEP (%)	155.83	64.35	79.44	80.63	42.71	84.59
	Seq2Point	MAE (W)	13.09	65.38	46.42	23.00	14.71	32.52
		NEP (%)	133.37	78.22	110.63	148.43	49.74	104.08
80%	Proposed	MAE (W)	9.90	**50.95**	**26.62**	**12.19**	**10.54**	**22.04**
		NEP (%)	101.00	**60.96**	**63.45**	**78.64**	**35.66**	**67.94**
	CRNN-Strong	MAE (W)	**9.55**	52.03	33.09	12.53	15.33	24.51
		NEP (%)	**97.00**	62.25	78.86	80.88	51.86	74.17
	Seq2Point	MAE (W)	17.11	53.98	29.60	14.61	13.17	25.69
		NEP (%)	174.23	64.58	70.55	94.28	44.53	89.63
100%	Proposed	MAE (W)	**10.89**	49.97	**30.40**	13.02	**11.82**	**23.22**
		NEP (%)	**110.96**	59.79	**72.46**	84.02	**39.97**	**73.44**
	CRNN-Strong	MAE (W)	12.70	**49.90**	37.14	**12.72**	14.85	25.46
		NEP (%)	129.34	**59.70**	88.52	**82.12**	50.24	81.98
	Seq2Point	MAE (W)	18.03	65.85	32.18	17.31	13.55	29.38
		NEP (%)	183.67	78.79	76.71	111.68	45.84	99.34
Average	Proposed	MAE (W)	**10.15**	**49.16**	**25.15**	**11.86**	**11.76**	**21.62**
		NEP (%)	**103.39**	**58.82**	**59.95**	**76.50**	**39.79**	**67.69**
	CRNN-Strong	MAE (W)	11.45	50.58	34.59	12.49	14.25	24.67
		NEP (%)	116.57	60.52	82.45	80.61	48.19	77.67
	Seq2Point	MAE (W)	14.94	63.49	37.75	16.47	19.83	30.49
		NEP (%)	152.15	75.96	89.96	106.29	67.07	98.29



(a) Microwave (40%).

(b) Fridge (20%).

(c) Dishwasher (20%).

(d) Washing Machine (20%).

(e) Kettle (20%).

Figure 6.2.: Ground-truth and estimated active power for the proposed and comparative methods for different percentages of strongly labeled data (shown in brackets).

Chapter 7.

Weakly Supervised Transfer Learning for Multi-label Appliance Classification

In the previous chapters, a novel supervision strategy has been introduced for NILM. The method has been firstly trained and evaluated on the same data domain. The method is proven effective but a more real-world scenario should be investigated. In fact, it is unlikely that training data are similar to data to be processed in the final environment. Based on the third experiment performed in Section 5 where the training set was composed of both data domains, good results have been obtained. This means that mixing the knowledge learnt from two domains can improve the generalization on both.

Starting from the promising performance, a transfer learning method will be introduced in this chapter. The method has been published in [3]. Transfer learning is an effective strategy for increasing generalization capability in these cases: recent methods operate by pre-training a neural network on a large dataset and then fine-tuning it on data acquired from the target environment [106, 96, 107]. However, this approach needs an additional acquisition phase in the target environment to collect the fine-tuning set. Semi-supervision can be involved but as reported in [108], although it is supposed that unlabeled data will bring a benefit, several empirical works demonstrated that the lack of labels decreased the performance.

In this view, weak labels are a trade-off between the complete absence of information on target data and the excessive effort requested to the user. In fact, in a target environment, weak labels can be obtained from the users' feedback, asking them if an appliance was active or not during a certain time window. Thus, an information is provided to the network although coarser. As pre-training set, public available data can be used or an already available pre-trained model can be considered. In this way, annotation collection is not necessary for the pre-training phase, and the user intervention is needed only to collect labels in the final environment. It is worth highlighting that the quan-

tity of annotated data required for transfer learning is much less than the data necessary to completely train a model from scratch. Thus, a transfer learning approach for multi-label appliance classification based on multiple-instance learning has been proposed for the first time. For the sake of completeness, different scenarios will be evaluated. In fact, weak labels are considered available also in the pre-training dataset. The benefit is the possibility to exploit all the types of labels available to enlarge as much as possible the training set. A subset of data with only partial information available is a real case. Instead of discarding them, they can be included in the training set.

7.1. Proposed Methodology

The formulation of the NILM problem as Multiple-Instance Learning refers to Section 2.1 and Section 5.1

Consider a training dataset $\mathcal{D}$ given by

$$\mathcal{D} = \mathcal{D}_{strong-weak} \cup \mathcal{D}_{weak}, \tag{7.1}$$

where $\mathcal{D}_{strong-weak} = \{(\mathbf{y}_1, \mathbf{w}_1, \mathbf{S}_1), \ldots, (\mathbf{y}_M, \mathbf{w}_M, \mathbf{S}_M)\}$ is a dataset composed of M bags annotated with strong and weak labels, and $\mathcal{D}_{weak} = \{(\mathbf{y}_{M+1}, \mathbf{w}_{M+1}), \ldots, (\mathbf{y}_{M+K}, \mathbf{w}_{M+K})\}$ is a dataset composed of K bags annotated with weak labels only.

The objective is to learn a function $\mathbf{f} : \mathbb{R}^L \to \mathbb{R}^{K \times L}$ from $\mathcal{D}$ that provides an estimate $\hat{\mathbf{S}}$ of $\mathbf{S}$ by using only the knowledge of the aggregate power $\mathbf{y}$. The function $\mathbf{f}(\cdot)$ is represented by a CRNN described in Section 5.1.1.

Similarly to [106], transfer learning here is performed by pre-training the neural network on a large dataset $\mathcal{D}^{(pt)}$, and then by fine-tuning it on a different dataset $\mathcal{D}^{(ft)}$. Both dataset can be composed as in Equation 7.1. $\mathcal{D}^{(pt)}$ can contain data only from the source domain or both from the source and target domains, while $\mathcal{D}^{(ft)}$ contains data only from the target domain.

As shown in Fig. 7.1, during fine-tuning, all the weights of the convolutional blocks are not updated (frozen) to avoid performance degradation [106]. Fine-tuning is performed only on the recurrent subpart and on the instance layer based on the performance obtained in the validation phase.

Based on the neural network architecture, two loss terms can be defined $\mathcal{L}_s$ and $\mathcal{L}_w$, respectively related to the instance and bag output. Both losses are the binary cross-entropy for the related output and they are calculated as in Equation 5.7 and Equation 5.8, where the bag index j has been omitted for simplicity of notation.

A significant advantage of the proposed method is that it allows to use strong or weak labels in the pre-training and fine-tuning phases depending on the

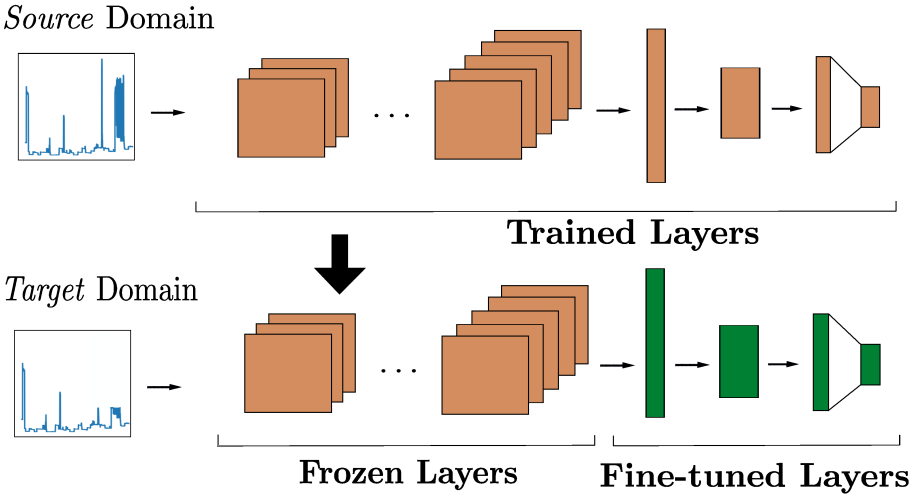


Figure 7.1.: Transfer learning with weak supervision. The model is trained with *source* domain data. Then the CNN blocks are frozen, and the remaining layers are fine-tuned with *target* domain data.

composition of $\mathcal{D}^{(pt)}$ and $\mathcal{D}^{(ft)}$. Supposing that

$$\mathcal{D}^{(pt)} = \mathcal{D}^{(pt)}_{strong-weak} \cup \mathcal{D}^{(pt)}_{weak}, \tag{7.2}$$

$$\mathcal{D}^{(ft)} = \mathcal{D}^{(ft)}_{strong-weak} \cup \mathcal{D}^{(ft)}_{weak}, \tag{7.3}$$

the model can be pre-trained both on strongly and weakly annotated data if $\mathcal{D}^{(pt)}_{strong-weak} \neq \varnothing$, or only on weakly annotated data if $\mathcal{D}^{(pt)}_{strong-weak} = \varnothing$. In the first case, the training loss is $\mathcal{L}_{tr} = \mathcal{L}_s + \lambda\mathcal{L}_w$, where λ balances the contribution of the two losses, while in the second case $\mathcal{L}_{tr} = \mathcal{L}_w$. Moreover, it is possible to combine data from different domains, e.g., $\mathcal{D}^{(pt)}_{strong-weak}$ can contain data from the source domain while $\mathcal{D}^{(pt)}_{weak}$ from the target domain.

Similarly, fine-tuning on the target domain data can be performed differently based on the available annotations: if $\mathcal{D}^{(ft)}_{strong-weak} \neq \varnothing$, fine-tuning is performed using both strongly and weakly labeled data and the related loss is $\mathcal{L}_{ft} = \mathcal{L}_s + \beta\mathcal{L}_w$, with β balancing the two losses. Conversely, if $\mathcal{D}^{(ft)}_{strong-weak} = \varnothing$, fine-tuning is performed only on weakly labeled data and $\mathcal{L}_{ft} = \mathcal{L}_w$.

It is worth noting that the case where $\mathcal{D}^{(pt)}_{strong-weak} \neq \varnothing$, and $\mathcal{D}^{(ft)}_{strong-weak} = \varnothing$ and $\mathcal{D}^{(ft)}_{-weak} \neq \varnothing$ is of particular relevance in a practical scenario since a large number of public datasets with strong annotations is available to pre-train the network, and the model can be fine-tuned by collecting data from the target domain and annotating it only with weak labels, thus reducing the labeling

Table 7.1.: Train, Validation and Test sets characteristics for REFIT. Number of labels is reported in thousands. SL: Strong Labels. WL: Weak Labels.

	Train			Validation			Test and Fine-tuning		
Appliance	Houses	Nr. of SL	Nr. of WL	Houses	Nr. of SL	Nr. of WL	Houses	Nr. of SL	Nr. of WL
KE	3, 5, 6, 7, 19	3217.0	54.1	3, 5, 6, 7, 19	678.6	12.7	2, 4, 8, 9	1182.9	24.4
MW	10, 12, 17, 19	2476.9	59.9	10, 12, 17, 19	606.9	9.9	4	436.6	8.4
FR	5, 9, 12	5433.7	62.0	5, 9, 12	1434.1	2.4	2, 15	1214.6	0.9
WM	5, 7, 15-18	1559.4	57.9	5, 7, 15-18	1788.2	4.1	2, 8, 9	1362.6	2.0
DW	5, 7, 13	520.292	52.7	5, 7, 13	2977.3	3.8	2, 9	2153.8	2.0
Nr. of bags	186.743			21.115			25.452		

effort.

7.2. Experimental Setting

7.2.1. Dataset

UK-DALE and REFIT datasets have been used to evaluate the performance of the proposed method. Kettle (KE), Microwave (MW), Fridge (FR), Washing Machine (WM), and Dishwasher (DW) are the appliances of interest. For both datasets characteristics please refers to Section 1 For training and validation sets composition please refers to Table 5.1. Table 7.1 report the details about sets for the two sets of bags created respectively from REFIT. Data was standardized using mean and standard deviation estimated on the training set.

7.2.2. Experimental setup

The experimental setup has been designed to evaluate several possible real-world scenarios that differ in data and annotations availability, based on the formulation in Section 8.1. The performance has always been evaluated on 70% of the REFIT "Test and Fine-tuning" set reported in Table 7.1.

Referring to Equation 7.2, three pre-training dataset compositions are defined:

1. Only weakly labeled data is available: in this case, $\mathcal{D}^{(pt)}_{strong-weak} = \varnothing$ and $\mathcal{D}^{(pt)}_{weak} \neq \varnothing$ is composed of bags from the UK-DALE dataset. Pre-training and test data in this case are from different domains.

2. Both strongly and weakly labeled data from the same domain is available: in this case, $\mathcal{D}^{(pt)}_{strong-weak} \neq \varnothing$ and $\mathcal{D}^{(pt)}_{weak} \neq \varnothing$, and they are both composed of bags from the UK-DALE dataset. As in the previous condition, pre-training and test data are from different domains.

3. Both strongly and weakly labeled data is available, but in this case they are from different domains: $\mathcal{D}^{(pt)}_{strong-weak} \neq \varnothing$ is composed of bags from

Table 7.2.: CRNN hyperparameters after tuning.

Dataset	H	U	K_e	p	CS
S-W UK-DALE	3	64	5	0.1	No
W UK-DALE	4	16	5	0.1	Yes
S-W REFIT	4	64	5	0.1	No

the UK-DALE dataset and $\mathcal{D}^{(pt)}_{weak} \neq \varnothing$ from bags of the REFIT dataset. Part of the pre-training and the test data are from the same domain.

Regardless the pre-training condition, the validation set is represented by UK-DALE.

Fine-tuning has been performed on 30% of the bags from each house of the REFIT "Test and Fine-tuning set" reported in Table 7.1. Referring to Equation 7.3, the fine-tuning dataset can be composed of 1) strongly and weakly annotated data from the target environment ($\mathcal{D}^{(ft)}_{strong-weak} \neq \varnothing$); 2) Only weakly labeled data from the target environment ($\mathcal{D}^{(ft)}_{weak} \neq \varnothing$). The last condition is of particular interest since it considers the case where data from the target environment is annotated only with weak labels. Thus, the labeling effort related to data collected in the target environment is significantly reduced.

Fine-tuned models have been compared to a baseline model (denoted as Baseline) obtained by using REFIT strongly and weakly labeled data both for training and validating the network and without fine-tuning it. The Baseline model, thus, considers the case where data from the same domain of the test set is available for training and represents an ideal case. Moreover, also the performance of pre-trained models are evaluated prior to fine-tuning (denoted as No Fine-Tuning).

For each pre-training condition, the Hyperband algorithm [99] from Keras tuner has been used to select the hyperparameters values that achieve the highest performance on the validation set. Learning is performed by using Adam [94] and the learning rate was fixed to 0.002. The number of filters F is set to 32. The final hyperparameters values are reported in Table 7.2. When the source dataset is only weakly labeled, fine-tuning the bidirectional and instance layers has proven the best performing method on the validation set. For the other two conditions, only the instance layer has been fine-tuned.

The threshold for binarizing instance level predictions has been determined on the validation set for each pre-training condition.

The code related to this work is available on GitHub[1].

[1] https://github.com/GiuTan/WeaklyTransferNILM

Table 7.3.: Results related to the Baseline and all the pre-training scenarios. Best results are reported in bold. Kettle: KE, Microwave: MW, Fridge: FR, Washing Machine: WM, Dishwasher: DW.

Method	**Labels**	**Dataset**	**KE**	**MW**	**FR**	**WM**	**DW**	F_1**-micro**
Baseline	Strong & Weak	REFIT	0.82	**0.82**	0.20	0.71	0.77	0.69
No Fine-Tuning	Weak	UK-DALE	0.68	0.44	0.01	0.45	0.33	0.41
Fine-Tuning	Strong & Weak	REFIT	**0.87**	0.72	0.22	0.68	0.71	0.68
Fine-Tuning	Weak	REFIT	0.66	0.57	0.00	0.49	0.36	0.45
No Fine-Tuning	Strong & Weak	UK-DALE	0.82	0.46	0.12	0.74	0.74	0.59
Fine-Tuning	Strong & Weak	REFIT	**0.87**	0.71	0.18	**0.76**	0.74	0.67
Fine-Tuning	Weak	REFIT	0.83	0.74	0.16	**0.76**	**0.78**	**0.71**
No Fine-Tuning	Strong & Weak	Mixed	0.78	0.46	0.11	0.42	0.62	0.52
Fine-Tuning	Strong & Weak	REFIT	0.86	0.81	**0.37**	0.62	0.68	0.67
Fine-Tuning	Weak	REFIT	0.85	0.77	0.25	0.44	0.69	0.61

7.3. Results and Discussion

Table 7.3 shows the results obtained with the three pre-training conditions and the related fine-tuning. The metrics used to evaluate the proposed approach is the F_1-score (F_1) and the related micro-average already defined in Equation 5.9 and Equation 5.10.

Observing the results, it is evident that weak supervision and transfer learning play a key role in obtaining performance close to the Baseline while reducing labeling effort. When the network is trained on weakly labeled data, and it is not fine-tuned (second row of Table 7.3), the performance is significantly lower compared to the Baseline. After fine-tuning with strong and weak labels and weak labels only, the F_1-micro increases by 65.8% and 8.9% (F_1:0.68 and F_1:0.45 compared to F_1:0.41), respectively. Compared to the Baseline, the F_1-scores of Kettle and Fridge improve by 6.1% and 9% (F_1:0.87 and F_1:0.22), respectively, if strong labels are considered in the fine-tuning set.

When the CRNN is trained on strong and weak labels from UK-DALE (fifth row of Table 7.3), F_1-micro is lower compared to the Baseline, while the F_1-score of the Washing Machine improves by 4%. Differently, compared to the model pre-trained only on weakly labeled data, all the appliances are better classified. After fine-tuning, the performance increases independently of the type of annotations. Remarkable performance can be observed when the CRNN is fine-tuned with weakly labeled data only (seventh row of Table 7.3), with an improvement of 20.3%. Moreover, the result improves by 2.9% for F_1-micro if compared to the Baseline.

This result is particularly significant since it means that pre-training the network with strong and weak labels from the source domain (UK-DALE) and fine-tuning it on weakly labeled data from the target domain (REFIT) results in superior performance compared to training on strongly and weakly labeled data from the target domain (REFIT). Single appliance behavior differs since

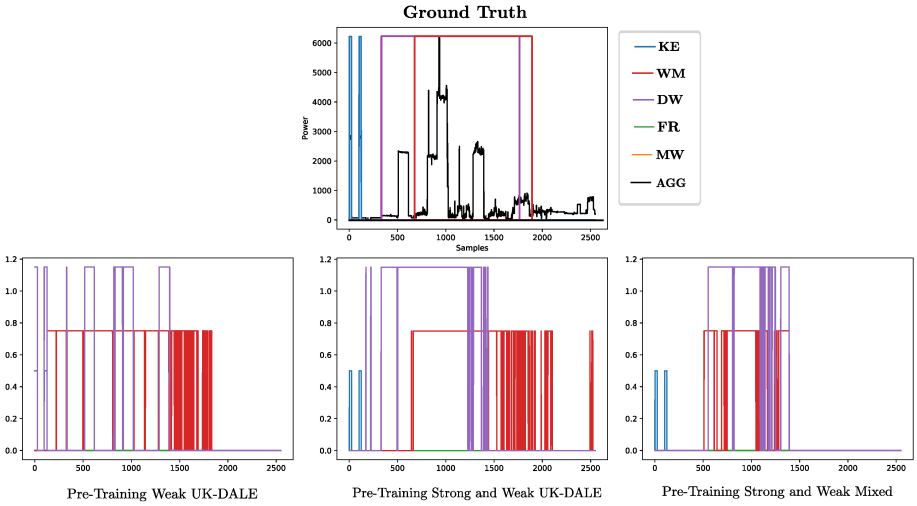


Figure 7.2.: Classification predictions produced by each pre-trained model after fine-tuning with weakly labeled data. Data is from REFIT house 2. AGG: Aggregate.

Kettle and Fridge show the best performance when fine-tuning is performed on strong and weak labels. At the same time, Microwave and Dishwasher are better classified when fine-tuning is performed on weak labels only. For the Washing Machine, in both conditions the performance improves compared to the Baseline by 7% after fine-tuning, meaning that weak labels are useful as well as the strong ones. The F_1-score of the Dishwasher improves by 1.2% with respect to the Baseline.

Without fine-tuning, pre-training the model on strong and weak annotations of the mixed dataset (eighth row of Table 7.3) obtains better performance compared to pre-training only on weak labels (second row of Table 7.3), but lower than pre-training on strong and weak annotations (fifth row of Table 7.3). This is related to the number of strongly annotated bags which are 20% of the entire UK-DALE set. When fine-tuned with strong annotations, the F_1-micro improves by 30%, while when target data are weakly labeled it improves by 17.3%. The Dishwasher is classified better when fine-tuning is performed with weak data only. In this condition, the labeling effort is reduced both for the pre-training and the fine-tuning set. Compared to the baseline, Kettle and Fridge F_1-micro improve by 4.8% and 85%, respectively. In summary, when the model is pre-trained only on weakly annotated data of the same domain, strong labels are required to fine-tune the network and obtain satisfactory performance. Conversely, when the network is pre-trained with strong and weak labels, weak data from the target environment are sufficient to improve per-

formance in terms of F_1-micro. For some appliances, by using weak labels for fine-tuning is better than using strong labels to improve performance. In the mixed case, since the quantity of strongly labeled data in the pre-training set is small, strong and weak fine-tuning results better than using only weak data.

Figure 7.2 shows the predictions produced by each pre-trained model after fine-tuning with weak labels. Since some predictions are overlapped, outputs are re-scaled for better visualization. It is evident that the fine-tuning on weak labels when pre-training is performed on strong and weak labels results in a more accurate prediction compared to the other models. In particular, this is highlighted for the Kettle and the Dishwasher.

Chapter 8.

A Weakly Supervised Active Learning Framework for Non-Intrusive Load Monitoring

While weak labels and transfer learning alleviate the burden of manual labeling, the process of fine-tuning still requires labeling a substantial number of data. The effort can be demanding for users.

Active Learning (AL) approaches [109, 110] are employed to optimize data selection for artificial intelligence algorithms. These approaches focus on identifying the most informative data, thereby reducing the number of labeled data segments that need to be added to the training dataset. Importantly, this reduction in data labeling does not compromise algorithm performance [83].

AL for NILM has not been extensively investigated yet - there have only been a few attempts for event-based methods using high-frequency load measurements, as already treated in Chapter 4.

Integrating the inexact supervised learning strategy into the AL framework with transfer learning avoids the need for expert labelling of target domain data, and annotation effort is reduced both in terms of the number of signal segments and the amount of information requested from users.

To address this gap, in this chapter a weakly supervised AL NILM approach is developed to reduce the number of signals that need to be weakly labeled by users. By asking to assign only weak labels to the most uncertain segments of the aggregate signal and sampling the fine-tuning set, the user annotation effort is further reduced while obtaining improved performance compared to [3, 88] upon which it is built. The proposed method is completely based on weak supervision, from the network pre-training to the adaptation in the target environment through to the AL procedure. This method has been accepted for publication in the journal "Integrated Computer-Aided Engineering" in 2024. [4].

8.1. Learning Strategy

Consider a pre-training dataset $\boldsymbol{D}^{(pt)}$ given by

$$\boldsymbol{D}^{(pt)} = \boldsymbol{D}^{(pt)}_{strong-weak} \cup \boldsymbol{D}^{(pt)}_{weak}, \tag{8.1}$$

where

$$\boldsymbol{D}^{(pt)}_{strong-weak} = \{(\mathbf{y}_1, \mathbf{w}_1, \mathbf{S}_1), \ldots, (\mathbf{y}_M, \mathbf{w}_M, \mathbf{S}_M)\}$$

is a dataset composed of M bags annotated with strong and weak labels, and

$$\boldsymbol{D}^{(pt)}_{weak} = \{(\mathbf{y}_{M+1}, \mathbf{w}_{M+1}), \ldots, (\mathbf{y}_{M+B}, \mathbf{w}_{M+B})\}$$

is a dataset composed of B bags annotated with weak labels only, from the source domain. Another set $\boldsymbol{D}_U = \{\mathbf{y}_{M+B+1}, \ldots, \mathbf{y}_{M+B+E}\}$ of electricity load measurements, called query pool, composed of E unlabelled bags is collected in the target environment, representing the pool for the AL process.

Based on the neural network architecture, two loss terms are defined $\mathcal{L}_s$ and $\mathcal{L}_w$, respectively, related to the instance and bag output. Both losses are the Binary Cross-Entropy functions for the related output calculated as is Equation 5.7 and Equation 5.8. Learning is initially performed by pre-training the neural network on a large public dataset $\boldsymbol{D}^{(pt)}$. A significant advantage of the proposed method is that it allows to use strong or weak labels in the pre-training phase depending on the composition of $\boldsymbol{D}^{(pt)}$. The model is pre-trained both on strongly and weakly annotated data if $\boldsymbol{D}^{(pt)}_{strong-weak} \neq \varnothing$, or only on weakly annotated data if $\boldsymbol{D}^{(pt)}_{strong-weak} = \varnothing$. In the first case, the training loss is $\mathcal{L}_{pt} = \mathcal{L}_s + \lambda\mathcal{L}_w$, where λ balances the contribution of the two losses, while in the second case it is $\mathcal{L}_{pt} = \mathcal{L}_w$. During fine-tuning, the weights of all the convolutional blocks are not updated (i.e., they are frozen) to avoid performance degradation [106]. Instead, fine-tuning is performed only on the recurrent subpart and on the instance layer using the dataset $\boldsymbol{Q}_{tot,i}$. This dataset contains a set of bags, annotated only with weak labels, obtained by labelling a subset $\boldsymbol{Q}_{U,i}$ of $\boldsymbol{D}_U$ at each iteration i. Additional details on $\boldsymbol{Q}_{tot,i}$ and $\boldsymbol{Q}_{U,i}$ are provided in Section 8.2. The fine-tuning loss $\mathcal{L}_{ft}$ is equal to $\mathcal{L}_w$ since only weak labels are supposed to be available from the target environment (i.e., $\boldsymbol{Q}_{tot,i}$ is annotated only with weak labels).

8.2. Weakly Supervised AL Framework

The proposed Weakly Supervised AL framework, schematically illustrated in Figure 8.1, comprises the CRNN model pre-trained using $\boldsymbol{D}^{(pt)}$, the query pool

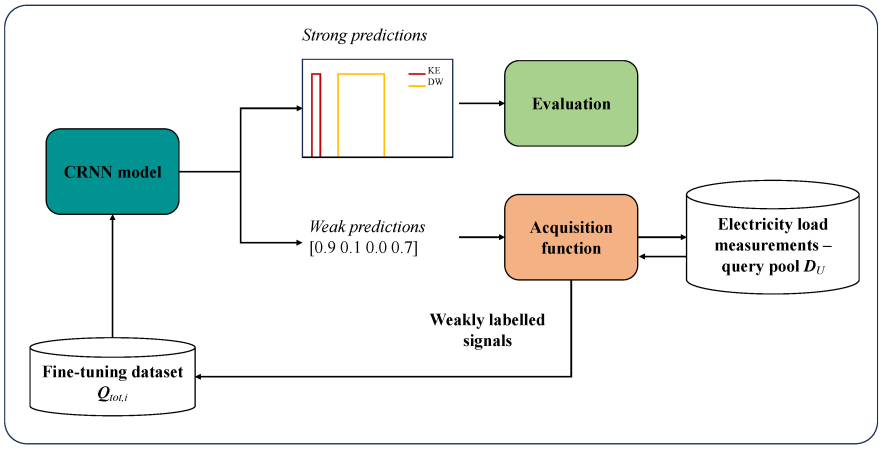


Figure 8.1.: Weakly Supervised AL Scheme. Each block corresponds to an element of the framework. The Convolutional Recurrent Neural Network (CRNN) model generates both strong and weak predictions. During the AL process, strong predictions are used to evaluate the current model, while weak predictions serve as input for the acquisition function. The acquisition function selects the windows to be labelled based on the uncertainty of the network predictions. The most uncertain windows are chosen, suggested to the user for annotation, and then incorporated into the fine-tuning set for the subsequent fine-tuning phase. A detailed description of the entire framework can be found in Section 8.2.

$\boldsymbol{D}_U$ for which only weak labels can be obtained on demand, and an acquisition function $q(\cdot)$ used to rank bags from $\boldsymbol{D}_U$ and choose the most informative ones to be included in the fine-tuning of the model.

The AL process is iterative, and the iterations are indicated with i and with $\mathbf{f}_{\boldsymbol{\theta}_i}$ the CRNN model trained at iteration i. The pre-trained model $\mathbf{f}_{\boldsymbol{\theta}_0}$ first makes predictions on the whole query pool $\boldsymbol{D}_U$ and provides its predictions to the acquisition function. The acquisition function then chooses a subset $\boldsymbol{Q}_{U,i} \subset \boldsymbol{D}_U$, with $i = 1, ..., I$ indexing the current query of most informative aggregate bags, accounting for model uncertainty when making predictions; the more uncertain the model is about a bag, the more the bag contributes towards the model prediction if included in fine-tuning.

Then, labels are queried for the chosen subset of bags as a result of the acquisition function. Let $\boldsymbol{Q}_{weak,i}$ be the weakly annotated set during the i-th query, composed of P bags. At the end of the loop, the model is fine-tuned

Algorithm 1 Pseudo-code for the Weakly Supervised AL procedure.

$i \leftarrow 1$
$\mathbf{f}_{\boldsymbol{\theta}_0}$: pre-trained CRNN model
$q(\cdot)$: acquisition function
$\boldsymbol{D}_U$: query pool, unlabelled
P: batch size
$\boldsymbol{Q}_{tot,i} \leftarrow \varnothing$
while $\mid \boldsymbol{D}_U \mid > 0$ **do**
  $\boldsymbol{Q}_{U,i} \leftarrow q(\mathbf{f}_{\boldsymbol{\theta}_{i-1}}, P, \boldsymbol{D}_U)$
  $\boldsymbol{D}_U \leftarrow \boldsymbol{D}_U \setminus \boldsymbol{Q}_{U,i}$
  $\boldsymbol{Q}_{weak,i} \leftarrow$ weakly labelled $\boldsymbol{Q}_{U,i}$
  $\boldsymbol{Q}_{tot,i} \leftarrow \boldsymbol{Q}_{tot,i-1} \cup \boldsymbol{Q}_{weak,i}$
  $\mathbf{f}_{\boldsymbol{\theta}_i} \leftarrow \mathbf{f}_{\boldsymbol{\theta}_0}$ fine-tuned with $\boldsymbol{Q}_{tot,i}$
  $i \leftarrow i + 1$
end while

using bags belonging to

$$\boldsymbol{Q}_{tot,i} = \boldsymbol{Q}_{tot,i-1} \cup \boldsymbol{Q}_{weak,i}, i = 1, ..., I, \tag{8.2}$$

queried up to the i-th query. Note that $\boldsymbol{Q}_{tot,0}$ is an empty set. The knowledge of the new, improved model $\mathbf{f}_{\boldsymbol{\theta}_i}, i > 0$ is used to further select samples for labelling. This procedure runs iteratively until all bags from the query pool are exhausted.

A pseudo-code of the weak AL procedure proposed in this paper is given in Algorithm 1.

Upon completion of the process, only the model that meets the desired criteria—namely, achieving a balance between good performance and a small data footprint, is employed to classify appliances. Importantly, the predictions made by intermediate models during the process are not considered. These post-fine-tuning models play a crucial role in selecting the subsequent batch of data for further refinement. Once this selection process is complete, the model can be safely discarded, as it will not be utilized in subsequent iterations.

The challenge of active learning (AL) with weak labels for a multi-appliance Non-Intrusive Load Monitoring (NILM) model is multifaceted. First, the limitation of having only weak labels available from the target domain. Additionally, the approach aims to monitor multiple appliances concurrently within the same network. However, this simultaneous monitoring can pose difficulties. Improving the performance of one appliance type does not necessarily translate to improvements for all other devices. In fact, enhancing the performance of one appliance may inadvertently lead to decreased performance for others.

This behavior significantly impacts the AL process, especially when dealing

with an appliance that exhibits notably lower performance compared to other loads in the household. In such cases, the selection of training instances (referred to as "bags") is more likely to focus on improving the most problematic appliance rather than addressing all appliances simultaneously. In the following sections, the strategy to tackle these challenges will be described.

8.3. Acquisition function

Acquisition function $q(\cdot)$ is used to rank bags in $\mathbf{D}_U$ with respect to their informativeness, choosing the best subset $\mathbf{Q}_U$ to include in model fine-tuning.

The acquisition function used in this paper is uncertainty-based, which demonstrated in [88] to be the best performing among several compared acquisition functions. In iteration $i, i > 0$, bags with the highest uncertainty levels, $\boldsymbol{Q}_{U,i} \subset \boldsymbol{D}_U$ are chosen to be labelled, denoted as $\boldsymbol{Q}_{weak,i}$, and included in fine-tuning dataset $\boldsymbol{Q}_{tot,i}$.

Weak level prediction of the model for a given bag is a vector containing probabilities of each appliance being in an active state inside that bag, which can be used to estimate uncertainty levels of the model. If a probability for a particular appliance is higher than decision threshold β then the model predicts that the appliance was active during the bag time period. The closer the prediction $\hat{w}_k$ of the model for an appliance k to β is, the more uncertain the model is about activation of this appliance, and the closer $\hat{w}_k$ to 1 or 0, the more certain the model is. Formally, an estimate of model uncertainty is defined as:

$$\delta_k[j] = \begin{cases} \hat{w}_k[j] & \hat{w}_k[j] < \beta \\ 1 - \hat{w}_k[j] & \hat{w}_k[j] \geq \beta \end{cases} \tag{8.3}$$

with $\delta_k[j]$ being the estimated uncertainty of the model for bag j for single appliance k, and $\hat{w}_k[j]$ is the model output, i.e., the model's estimated probability that k-th appliance was active in the bag j.

Since the problem considered in this paper is multi-label classification, with multiple appliances considered at the same time, two ways of estimating the overall model uncertainty $\delta[j]$ for bag j are:

- by taking maximum uncertainty level across appliances present in the house:

$$\delta[j] = \max_k \delta_k[j] \tag{8.4}$$

- by averaging uncertainty level over all appliances present in the house:

$$\delta[j] = \frac{1}{\tilde{K}} \sum_{k=1}^{\tilde{K}} \delta_k[j]. \tag{8.5}$$

Table 8.1.: Uncertainty-based acquisition function example: Uncertainty levels for each appliance are calculated as per (8.3), and, maximum or mean uncertainty values are calculated based on (8.4) and (8.5), respectively. In this example, a batch of $P = 4$ most uncertain bags is chosen.

Bag index	Weak level prediction $\hat{w}_k[j]$				Uncertainty $\delta_k[j]$				Maximum	Mean
j	KE	MW	WM	DW	KE	MW	WM	DW	uncertainty	uncertainty
0	0.1	0.6	0.4	0.8	0.1	0.4	0.4	0.2	0.4	0.275
1	0.2	0.85	0.33	0.68	0.2	0.15	0.33	0.32	0.33	0.25
2	0.99	0.2	0.87	0.3	0.01	0.2	0.13	0.3	0.3	0.16
3	0.56	0.38	0.25	0.92	0.44	0.38	0.25	0.08	0.44	0.2875
4	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
5	0.67	0.43	0.01	0	0.33	0.43	0.01	0	0.43	0.1925
6	0.36	0.15	0.64	0.75	0.36	0.15	0.36	0.25	0.36	0.28
7	0.83	0.72	0.59	0.41	0.17	0.28	0.41	0.41	0.41	0.3175
8	0	0.5	0	0	0	0.5	0	0	0.5	0.125
9	0.04	0.99	0.88	0.02	0.04	0.01	0.12	0.02	0.12	0.0475

Then, the set of bags $\boldsymbol{Q}_{U,i}$ with the highest uncertainty $\delta[j]$ is included in the fine-tuning set. The resulting acquisition function, $q(\cdot)$, is as described in Algorithm 2.

Algorithm 2 Acquisition function

$\mathbf{f}_i$: CRNN model
$\boldsymbol{D}_U$: query pool, unlabelled
P: batch size
procedure $q(\mathbf{f}_i, P, \boldsymbol{D}_U)$
  for j in $\{1, ..., \mid \boldsymbol{D}_U \mid\}$ **do**
    $\hat{\mathbf{w}}[j] \leftarrow \mathbf{f}_i(\boldsymbol{D}_U[j])$
    calculate uncertainty $\delta[j]$
  end for
  $ind = argsort([\delta[1]...\delta[\mid \boldsymbol{D}_U \mid]], descend.)[: P]$
  return $\boldsymbol{D}_U[ind]$
end procedure

A toy example of how the acquisition function described above works, for both cases of maximising and averaging uncertainties of individual appliances is given in Table 8.1. Table 8.1 shows the selected bags (a batch of $P = 4$) in grey for maximum uncertainty across all appliances in the 4-th column and for maximum average uncertainty over all appliances in the 5-th column.

The code used to implement the approach is available on Github[1].

[1] https://github.com/GiuTan/WeaklySupervisedActiveLearning-for-NILM

Table 8.2.: Fine-Tuning and Test sets characteristics for REFIT. Number of labels is reported in thousands. WL: Weak Labels.

Appliance	Nr. of WL			
	House 2	House 4	House 5	House 19
KE	2.9	12	9.5	13.6
MW	-	12	-	13.6
WM	2.9	-	0.5	-
DW	2.9	-	0.5	-
Nr. of bags	2.9	12	9.5	13.6

8.4. Experimental Setting

8.4.1. Dataset

UK-DALE [24] and REFIT [25] datasets are used to evaluate the performance of the proposed method with typical appliances present in most households - Kettle (KE), Microwave (MW), Washing Machine (WM), and Dishwasher (DW). The fridge is excluded in this work. This decision was made since a fridge is typically always in operation, which would mean the user would consistently assign the ON label. Although the fridge is not monitored, it is present in the aggregate dataset.

Datasets have been used to create two sets of bags, one with UK-DALE data from Houses 1, 3 and 5 and one with data from four REFIT Houses 2, 4, 5 and 19. This choice has been made to include 4 houses that have different aggregate consumption characteristics, and have at least two appliances present in each house for evaluation. Note that the occurrence of appliance activations and the number of strong labels associated with each appliance in both sets of bags are balanced. Table 5.1 reports the details about training set for UK-DALE. Table 8.2 reports validation, and test sets details for REFIT. The set used to validate the performance during AL process is the test set. Data was standardised subtracting the mean and dividing by the standard deviation, estimated these values on the pre-training set.

8.4.2. Experiments setup

The experimental setup has been designed to evaluate several possible real-world scenarios that differ in annotation availability, based on the formulation in Section 8.1. In this way, the benefits from the AL procedure in more pre-training conditions can be evaluated. The performance has always been evaluated on 70% of the REFIT "Test and Fine-tuning" set reported in Table 8.2.

Referring to Equation 8.1, two pre-training dataset compositions are defined:

- Scenario 1: only weakly labelled data is available: in this case, $\boldsymbol{D}^{(pt)}_{strong-weak} = \varnothing$ and $\boldsymbol{D}^{(pt)}_{weak} \neq \varnothing$ is composed of bags from the UK-DALE dataset.
- Scenario 2: both strongly and weakly labelled data from the same domain are available: in this case, $\boldsymbol{D}^{(pt)}_{strong-weak} \neq \varnothing$ and $\boldsymbol{D}^{(pt)}_{weak} \neq \varnothing$, and they are both composed of bags from the UK-DALE dataset.

Regardless of the pre-training condition, the validation set is represented by UK-DALE.

The bags that populate the query pool $\mathbf{D}_U$ for AL and that are used for the fine-tuning are up to 30% of the bags from each house of the REFIT "Test and Fine-Tuning set", reported in Table 8.2.

For each pre-training condition, the Hyperband algorithm [99] from Keras tuner has been used to select the hyperparameters values that achieve the highest performance on the validation set. During the AL process, there is not any optimisation of hyperparameters. This is because the structure of the fine-tuned network is the same as that of the pre-trained network. The pre-trained network has already been optimised during the pre-training phase, performed in [3]. Adam [94] is used as optimiser and the learning rate was fixed to 0.002 and F to 32. In the experiments $L = 2550$ (that is a window of 4.15 hours) samples is set for the bag dimension and $P = 64$ is the batch size.

When the source dataset is only weakly labelled, fine-tuning the bidirectional and instance layers has proven the best performing method on the validation set. When strongly labelled data are also available, only the instance layer has been fine-tuned.

8.4.3. Benchmark method

In [3] a weakly supervised transfer learning approach has been proposed. Both the pre-training and the fine-tuning exploits only weak labels, or both weak and strong labels. In the fine-tuning phase, a set of weakly annotated signals has been supplied to the network to adapt the pre-trained model on the target environment domain. The best models obtained from the proposed method have been compared to "No Fine-Tuning" model [3], thus prior to fine-tuning, and "Weak Transfer Learning" model [3] obtained using the complete set of query pool data weakly annotated.

Additionally, the proposed method is compared against a semi-supervised method based on knowledge distillation, proposed in [5], that is pre-trained using only strong labels, but in the fine-tuning phase only unlabelled data is fed to the model, as considering that labels from the target environment are not readily available. Thus, unlabelled data are exploited during learning with the same strategy adopted in the original work that proposed a semi-supervised

approach [5]. Because of absence of labels from the target environment, and the way that the model works, bags with the lowest uncertainty were chosen instead of the highest during the AL process for this benchmark.

8.4.4. Evaluation metrics

Two metrics have been used to evaluate the proposed approach. The first is the F_1-score (F_1) and the related micro-average already defined in Equation 5.9 and Equation 5.10.

Optimal point of AL iteration process is determined as a point at iteration i with F_1-score $F_{1,i}$ that has the minimum distance d_i from an "ideal" point - no data labelled, and perfect performance of $F_1 = 1$, as in [88]. The distance is calculated according to Equation (8.6), where $|\boldsymbol{Q}_{tot,i}|$ denotes the total number of bags queried up to iteration i, and $|\boldsymbol{Q}_{tot,I}|$ denotes the maximum number of bags that can be queried.

$$d_i = \sqrt{\left(\frac{|\,\boldsymbol{Q}_{tot,i}\,|}{|\,\boldsymbol{Q}_{tot,I}\,|}\right)^2 + (1 - F_{1,i})^2}. \tag{8.6}$$

8.5. Results

This section presents the results obtained from the two experimental scenarios, as well as from the semi-supervised benchmark method. F_1-scores are shown per appliance for each house. Models pre-trained on UK-DALE were transferred to REFIT houses 2, 4, 5 and 19 - Dataset column indicates the fine-tuning test set. The optimal points and maximum performances obtained during the AL process are given together with the percentage of query pool data labelled and added to the fine-tuning dataset to achieve that performance. Note that not all houses contain all the appliances - results are shown only for monitored appliances installed in the selected buildings.

8.5.1. Semi-supervised benchmark results

Experimental results for the semi-supervised benchmark approach [5] are presented in Table 8.3. In this case, strongly labelled data were used during the pre-training phase, and unlabelled data were utilised throughout the AL process. This scenario is challenging because with the semi-supervised strategy the model is fine-tuned with unseen data from the target environment without any labels provided. According to Table 8.3, the performance in House 2 does not improve after fine-tuning with all available data (100% of unlabelled bags used). There is a very limited improvement with AL for kettle only, but the performance level of the fine-tuning case with 100% of unlabelled bags used

Table 8.3.: Benchmark - semi supervised method [5]. Model is pre-trained using strong labels, but fine-tuned using only unlabelled data from target environment. Results of the proposed approach are shown in the following format: metric (% of activation samples added to fine-tuning dataset).

	Method	**KE**	**MW**	**WM**	**DW**	F_1**-micro**
H2	No Fine-Tuning [5]	0.55	-	0.41	0.58	0.50
	Unsupervised Transfer Learning [5]	0.55	-	0.41	0.58	0.50
	AL (max uncertainty) - optimal point	0.55 (13.3%)	-	0.41 (6.7%)	0.58 (6.7%)	0.50 (6.7%)
	AL (max uncertainty) - best F1	0.56 (80%)	-	0.41 (6.7%)	0.58 (6.7%)	0.50 (6.7%)
	AL (mean uncertainty) - optimal point	0.54 (13.3%)	-	0.41 (6.7%)	0.58 (6.7%)	0.50 (6.7%)
	AL (mean uncertainty) - best F1	0.56 (73.3%)	-	0.41 (6.7%)	0.58(6.7%)	0.50 (6.7%)
H4	No Fine-Tuning [5]	0.42	0.38	-	-	0.39
	Unsupervised Transfer Learning [5]	0.44	0.44	-	-	0.44
	AL (max uncertainty) - optimal point	0.44 (13.8%)	0.41 (10.3%)	-	-	0.42 (13.8%)
	AL (max uncertainty) - best F1	0.45 (20.7%)	0.44 (38%)	-	-	0.44 (38%)
	AL (mean uncertainty) - optimal point	0.45 (1.7%)	0.41 (12.1%)	-	-	0.41 (12.1%)
	AL (mean uncertainty) - best F1	0.45 (1.7%)	0.44 (98.2%)	-	-	0.44 (98.2%)
H5	No Fine-Tuning [5]	0.86	-	0.02	0.04	0.05
	Unsupervised Transfer Learning [5]	0.86	-	0.02	0.04	0.05
	AL (max uncertainty) - optimal point	0.86 (4.3 %)	-	0.02 (2.2 %)	0.04 (2.2 %)	0.05 (2.2 %)
	AL (max uncertainty) - best F1	0.87 (60.9 %)	-	0.02 (2.2 %)	0.04 (2.2 %)	0.05 (2.2 %)
	AL (mean uncertainty) - optimal point	0.86 (4.3 %)	-	0.02 (2.2 %)	0.04 (2.2 %)	0.05 (2.2 %)
	AL (mean uncertainty) - best F1	0.87 (97.8 %)	-	0.02 (2.2 %)	0.04 (2.2 %)	0.05 (2.2 %)
H19	No Fine-Tuning [5]	0.82	0.61	-	-	0.69
	Unsupervised Transfer Learning [5]	0.82	0.61	-	-	0.69
	AL (max uncertainty) - optimal point	0.82 (3.1 %)	0.63 (1.5 %)	-	-	0.70 (1.5 %)
	AL (max uncertainty) - best F1	0.82 (3.1 %)	0.64 (89.2 %)	-	-	0.70 (1.5 %)
	AL (mean uncertainty) - optimal point	0.82 (3.1 %)	0.62 (1.5 %)	-	-	0.69 (1.5 %)
	AL (mean uncertainty) - best F1	0.83 (43.1 %)	0.63 (60 %)	-	-	0.70 (60 %)

can be achieved using a smaller amount of data (6.7% - 13.3%). In House 4, performance improves when all available bags from target environment are used, and the amount of data can be reduced to at least 38% of all data. In house 5, the situation is similar as in house 2 - no improvement after fine-tuning with all available unlabelled bags, and only small improvement for kettle with large portion of unlabelled bags used with AL. There is a similar situation in house 19 - no improvement after fine-tuning with all available unlabelled data, but small improvement for microwave with AL. The results from this benchmarking scenario suggest that while some improvement can be achieved using only unlabelled data to fine-tune the model to the new environment, it is not sufficient, and adding some labelled data is desirable. Therefore, results for weakly supervised AL scenarios are presented next.

8.5.2. Weakly Supervised AL Performance

Experimental results for the scenario where only weakly labelled data is available in the pre-training phase - pre-training scenario 1, and weak labels are used throughout the AL process, are presented in Table 8.4. This scenario is very challenging, because the model never sees strong labels, neither during pre-training nor during fine-tuning phase.

In House 2, with weak transfer learning (100% bags labelled), performance increases compared to the one before fine-tuning (0% bags labelled) for dishwasher, but drops for kettle and washing machine due to over-fitting. However, for kettle, with AL when maximising uncertainty over appliances, performance increase is achieved at optimal AL point with 13.3% bags labelled, and when averaging uncertainty over appliances, performance increases with labelling 20% of bags, reducing labelling effort by 86.7% and 80% respectively. For washing machine, labelling 6.7% of bags retains performance whether uncertainty is maximised or averaged over appliances. For dishwasher, performance is increased at optimal point with only 13.3% of bags labelled with maximising, and with 6.7% when averaging uncertainty over appliances. Micro F_1-score is retained in all AL cases.

This situation is a consequence of different appliance signature characteristics - a kettle activation, as a short duration appliance, is more likely to be present in bags with other activations from other devices, and hence needs more queries to augment its learning to see sufficient kettle activations with different aggregates. Washing machine is likely to be confused with dishwasher and, hence, in the absence of strong labels its performance cannot be improved, especially for the low-power state. For dishwasher, there are more high power samples in one activation and, therefore, with more training samples in the weak labels, it is possible to improve.

Table 8.4.: Results - pre-training Scenario 1. Results of the proposed approach are shown in the following format: metric (% of activation samples added to fine-tuning dataset).

	Method	**KE**	**MW**	**WM**	**DW**	F_1**-micro**
H2	No Fine-Tuning [3]	0.73	-	0.62	0.70	0.67
	Weak Transfer Learning [3]	0.59	-	0.42	0.73	0.58
	Proposed (max uncertainty) - optimal point	0.74 (13.3%)	-	0.62 (6.7%)	0.71 (13.3%)	0.67 (6.7%)
	Proposed (max uncertainty) - best F_1	0.79 (73.3%)	-	0.62 (6.7%)	0.74 (33.3%)	0.67 (6.7%)
	Proposed (mean uncertainty) - optimal point	0.80 (20%)	-	0.62 (6.7%)	0.71 (6.7%)	0.67 (6.7%)
	Proposed (mean uncertainty) - best F_1	0.80 (20%)	-	0.62 (6.7%)	0.73 (20%)	0.67 (6.7%)
H4	No Fine-Tuning [3]	0.54	0.53	-	-	0.53
	Weak Transfer Learning [3]	0.59	0.65	-	-	0.63
	Proposed (max uncertainty) - optimal point	0.61(1.7%)	0.64 (1.7%)	-	-	0.63 (1.7%)
	Proposed (max uncertainty) - best F_1	0.61 (1.7%)	0.72 (67.2%)	-	-	0.65 (67.2%)
	Proposed (mean uncertainty) - optimal point	0.58 (8.8%)	0.63 (10.5%)	-	-	0.61 (10.5%)
	Proposed (mean uncertainty) - best F_1	0.60 (52.6%)	0.70 (66.7%)	-	-	0.65 (66.7%)
H5	No Fine-Tuning [3]	0.78	-	0.24	0.28	0.51
	Weak Transfer Learning [3]	0.79	-	0.32	0.28	0.55
	Proposed (max uncertainty) - optimal point	0.80 (2.2%)	-	0.30 (6.5 %)	0.27 (10.7 %)	0.56 (10.7 %)
	Proposed (max uncertainty) - best F_1	0.80 (2.2 %)	-	0.36 (95.6 %)	0.28 (50 %)	0.57 (54.3 %)
	Proposed (mean uncertainty) - optimal point	0.80 (2.2 %)	-	0.34 (26.1 %)	0.28 (4.3 %)	0.56 (6.5 %)
	Proposed (mean uncertainty) - best F_1	0.80 (2.2 %)	-	0.34 (26.1 %)	0.29 (52.2 %)	0.56 (6.5 %)
H19	No Fine-Tuning [3]	0.66	0.68	-	-	0.67
	Weak Transfer Learning [3]	0.75	0.69	-	-	0.71
	Proposed (max uncertainty) - optimal point	0.80 (3.1 %)	0.70 (1.5 %)	-	-	0.73 (1.5 %)
	Proposed (max uncertainty) - best F_1	0.81 (64.6 %)	0.71 (29.2 %)	-	-	0.73 (1.5 %)
	Proposed (mean uncertainty) - optimal point	0.78 (2.7 %)	0.70 (8.1 %)	-	-	0.73 (2.7 %)
	Proposed (mean uncertainty) - best F_1	0.79 (13.5 %)	0.71 (27 %)	-	-	0.74 (13.5 %)

Table 8.5.: Results - pre-training Scenario 2. Results of the proposed approach are shown in the following format: metric (% of activation samples added to fine-tuning dataset).

	Method	**KE**	**MW**	**WM**	**DW**	**F_1-micro**
H2	No Fine-Tuning [3]	0.78	-	0.78	0.84	0.82
	Weak Transfer Learning [3]	0.83	-	0.82	0.83	0.82
	Proposed (max uncertainty) - optimal point	0.82 (6.7%)	-	0.80 (6.7%)	0.83 (6.7%)	0.82 (6.7%)
	Proposed (max uncertainty) - best F_1	0.83 (13.33%)	-	0.82 (46.7%)	0.84 (93.3%)	0.82 (6.7%)
	Proposed (mean uncertainty) - optimal point	0.83 (6.7%)	-	0.80 (6.7%)	0.83 (6.7%)	0.82 (6.7%)
	Proposed (mean uncertainty) - best F_1	0.84 (86.7%)	-	0.82 (26.7%)	0.84 (33.3%)	0.83 (66.7%)
H4	No Fine-Tuning [3]	0.71	0.69	-	-	0.69
	Weak Transfer Learning [3]	0.73	0.73	-	-	0.73
	Proposed (max uncertainty) - optimal point	0.76 (6.9%)	0.84 (5.2%)	-	-	0.81 (5.2%)
	Proposed (max uncertainty) - best F_1	0.77 (14%)	0.86 (73.7%)	-	-	0.81 (5.2%)
	Proposed (mean uncertainty) - optimal point	0.78 (1.7%)	0.85 (1.7%)	-	-	0.83 (1.7%)
	Proposed (mean uncertainty) - best F_1	0.78 (1.7%)	0.86 (28.1%)	-	-	0.83 (1.7%)
H5	No Fine-Tuning [3]	0.94	-	0.20	0.43	0.60
	Weak Transfer Learning [3]	0.95	-	0.41	0.55	0.70
	Proposed (max uncertainty) - optimal point	0.96 (4.3%)	-	0.41 (26.1%)	0.54 (17.4%)	0.69 (17.4%)
	Proposed (max uncertainty) - best F_1	0.96 (4.3 %)	-	0.42 (76.1%)	0.57 (60.9%)	0.72 (65.2%)
	Proposed (mean uncertainty) - optimal point	0.96 (2.2 %)	-	0.36 (28.3 %)	0.51 (2.2 %)	0.67 (2.2%)
	Proposed (mean uncertainty) - best F_1	0.96 (2.1 %)	-	0.40 (39.1 %)	0.58 (28.3 %)	0.71 (28.3 %)
H19	No Fine-Tuning [3]	0.88	0.75	-	-	0.80
	Weak Transfer Learning [3]	0.76	0.69	-	-	0.71
	Proposed (max uncertainty) - optimal point	0.91 (7.7 %)	0.73 (1.5 %)	-	-	0.78 (1.5 %)
	Proposed (max uncertainty) - best F_1	0.94 (72.3 %)	0.73 (1.5 %)	-	-	0.78 (1.5 %)
	Proposed (mean uncertainty) - optimal point	0.89 (4.6 %)	0.76 (7.7 %)	-	-	0.80 (1.5 %)
	Proposed (mean uncertainty) - best F_1	0.89 (4.6 %)	0.76 (7.7 %)	-	-	0.81 (7.7 %)

In House 4, weak transfer learning (100% bags labelled) increases performance for both kettle and microwave, as well as the micro F_1-score. With weak AL, for kettle, at optimal point, performance increase is achieved with 1.7% and 8.8% bags labelled when maximising and averaging uncertainty over appliances, respectively, reducing labelling effort by 98.3% and 92.2%. For microwave, at optimal point performance is increased with 1.7% and 10.5% bags labelled when maximising and averaging uncertainty over appliances, respectively. Micro F_1-score increased at optimal points with only 1.7% and 10.5% bags labelled when maximising and averaging uncertainty over appliances, respectively.

Considering best F_1-score, kettle needs 52% additional samples for fine-tuning when considering mean uncertainty across appliances but only 1.7% more when considering maximum uncertainty. This is due to the fact that House 4 is much noisier in terms of unknown appliances present in the aggregate signal - it has noise to aggregate ratio (NAR [111]) of 0.91, with noise calculated as in [3], compared to the NAR value of house 2 which is 0.79. Microwave needs more additional samples due to its short activation time and high probability of activation in presence of other appliances, hence, the model requires more weakly labelled bags to improve.

In house 5, performance is poor before fine-tuning for washing machine and dishwasher. However, overall performance, as well as per-appliance performance, does improve (or remains the same for the dishwasher) with weak transfer learning (100% bags labelled), and also with weak AL with reduced amount of labelled data. With weak AL, the amount of data that needs labelling increases from 2.2 to 10.7 % when maximising uncertainty across appliances, and from 2.2 to 26.1 % when averaging, at optimal points. At best F_1-score, washing machine and dishwasher need significantly larger portion of labelled data, due to poor performance in the beginning. Consequently, micro F_1-score also peaks at higher percentage of data labelled. House 5 is noisier than House 2 as indicated by a NAR value of 0.84, but lower than House 4, hence it exhibits a good performance for kettle, but washing machine and dishwasher have more complex patterns which are different from device to device, so it is hard to improve them significantly with weak labels only for this house.

In House 19, performance improves with AL exceeding the performance of weak fine-tuning (100 % bags labelled), requiring only 1.5 - 3.2 % of bags to be weakly labelled when maximising, and 2.7 to 8.1 % when averaging uncertainty across appliances, at optimal points. NAR value of House 19 is the highest among all test houses - 0.93, but starting performance before any fine-tuning is good, which indicates that this domain has more similarities with training data than previous testing domains.

Table 8.5 shows results where strong and weak labels are used in the pre-

training phase - pre-training scenario 2, and weak labels are used in the AL phase. This scenario is more favourable compared to the previous one, because even though only weak labels are available during fine-tuning phase, strong labels are available in the pre-training phase.

Compared to Scenario 1, as expected, performance for all appliances in all houses is improved over the baseline [3] with significantly less additional fine-tuning data. This behaviour can be attributed to the inclusion of strong labels during the pre-training phase, which increased the network's knowledge, thereby necessitating a lesser quantity of data to achieve comparable or improved results.

Next, levels of uncertainty observed at the start of the AL process are discussed. In Scenario 1, weak labels only are present in the pre-training phase, and the model tends to be either overconfident or very unconfident (as shown by the uncertainty histogram in Fig. 8.2 (top) - most of bags have low uncertainty values - and lower uncertainty means higher confidence), and the performance before fine-tuning is not as good as with strong labels present (Scenario 2). On the other hand, when strong labels are present in the pre-training phase (Scenario 2), performance before fine-tuning is better, but there are not as many low uncertainty (high confidence) bags as in Scenario 1 (as shown in Figure 8.2). The model has been shown strong labels, hence better performance, but is also more uncertain (i.e., histogram is more flat) due to learning from strong labels with overlapping activations of multiple appliances contained in a bag. It is also worth noting that more high uncertainty bags are observed for kettle than for microwave. Uncertainty levels among bags that are queried for REFIT house 4 in each experimental scenario are shown in Figure 8.3: Scenario 1 with mean uncertainty across appliances – upper left; Scenario 1 with maximum uncertainty across appliances – upper right; Scenario 2 with mean uncertainty across appliances - lower left; and Scenario 2 with maximum uncertainty across appliances – lower right. The figures show uncertainty level of microwave (orange) stacked to uncertainty level of kettle (blue) for each bag queried in the beginning of the AL process, before any fine tuning. In case of using maximum uncertainty across appliances as overall uncertainty measure, the model tends to pick bags in which uncertainty is high for kettle, but not necessarily for microwave – according to histograms in Figure 8.2, kettle has more high uncertainty bags in general. On the other hand, if using mean uncertainty across appliances as overall uncertainty measure, bags are picked so that both appliances have high uncertainty. Therefore, as described in Section 4.3, querying based on mean uncertainty is more reliable and gives better overall improvement of the model.

From both Tables 8.4 and 8.5, it can be observed that with proposed optimal point (Eq 8.6), performance improvement (House 2: 1.2%, House 4: 14%, House

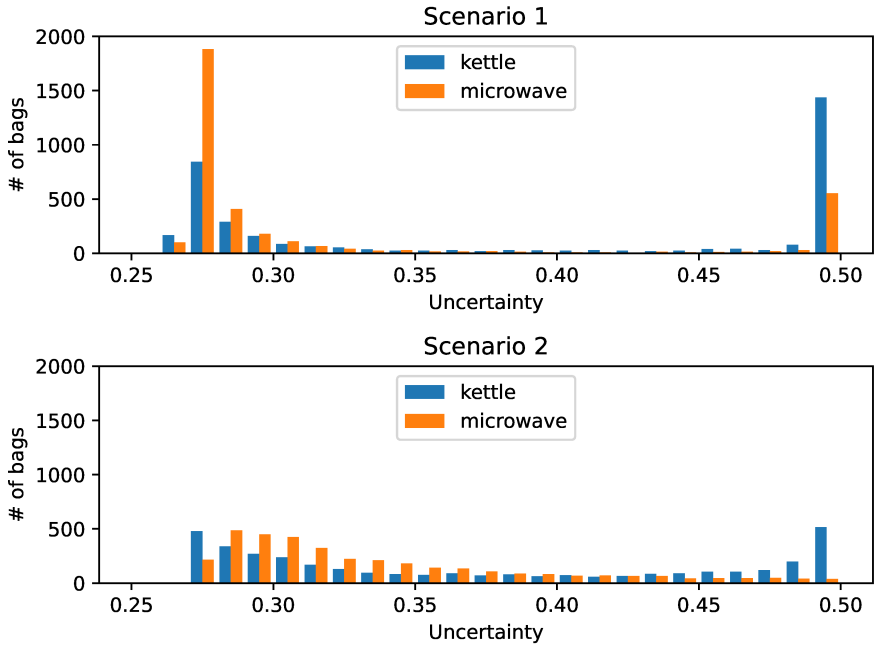


Figure 8.2.: Observed uncertainty levels in Scenario 1 (top) and Scenario 2 (bottom) for the whole query pool of House 4 bags.

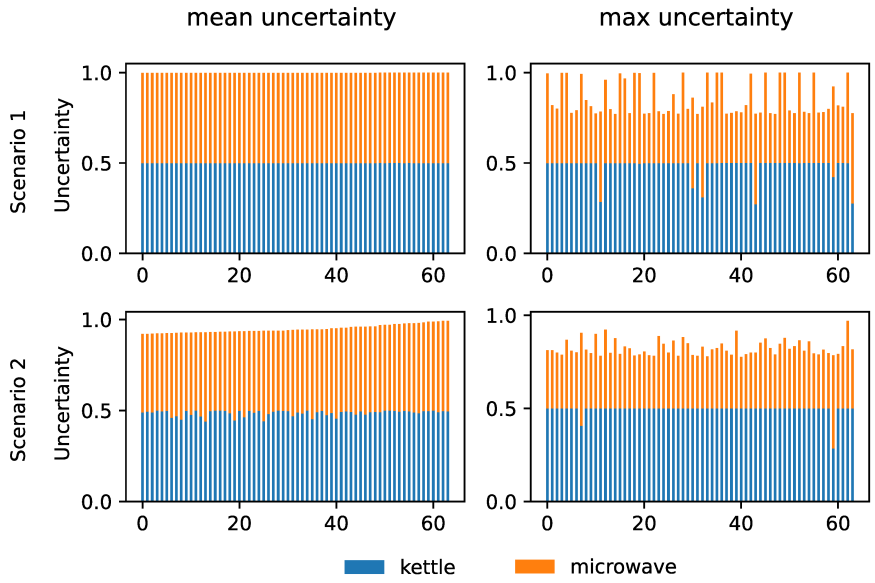


Figure 8.3.: Observed ratio of uncertainty between kettle and microwave in Scenarios 1 (top) and 2 (bottom), when using mean (left) and maximum (right) uncertainty across present appliances.

5: 2.9%, House 19: 14%), for both acquisition functions, is almost the same as best F1 performance, with significantly less additional fine-tuning data.

AL curve with optimal points marked obtained in House 4 with mean uncertainty over appliances is shown in Figure 8.4. In the beginning of the AL process, useful bags are chosen in the first couple of iterations, after which performance becomes steady for kettle, and improves further for microwave.

From the presented results, it is evident that sometimes adding less data is better than adding more, because not all data samples are useful, and not all data samples do improve the pre-trained model. Therefore, AL approaches can be used to select only high-uncertainty data and label and add only them to fine-tuning dataset. An important note is that *weak labels only* can be used throughout the AL process, and model performance can still improve. This is very encouraging, especially bearing in mind that weak labels are easily obtained, and that they could be obtained even from lay users, who do not have any knowledge of NILM and appliance signatures - weak labels could be inferred by only asking users when did they run specific device.

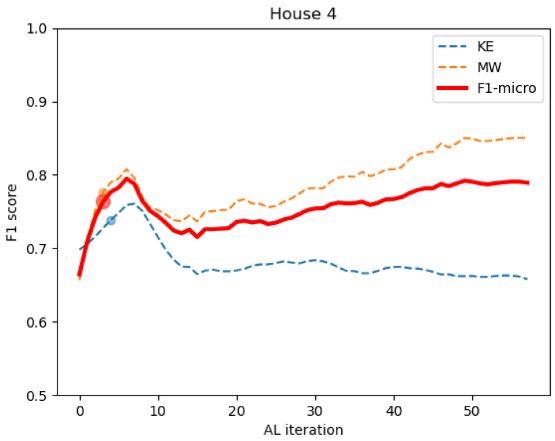


Figure 8.4.: AL curve obtained at REFIT House 4 in Scenario 2 when averaging uncertainty across present appliances. Original curve is smoothed using Savitsky-Golay filter of length 11 and order 3.

8.5.3. Complexity Details

In this section, a brief discussion on the complexity of the proposed approach is provided. It is worth noting that this framework is primarily designed for data efficiency without compromising performance, but the method itself does not focus on reducing computational complexity.

In each AL iteration, there are two phases that require significant computational resources: acquisition and fine-tuning phase. In the acquisition phase, the model needs to examine all signal bags belonging to the query pool and rank them by uncertainty, which has a complexity of $O(n^2)$. The cost of this step reduces as the AL process progresses because the size of the query pool decreases. The fine-tuning phase then uses acquired signal segments to fine-tune the model. The cost of this increases as the AL process progresses because the fine-tuning set size increases as newly queried signal segments are added. The CRNN model used in this paper consumes 976.28 kB of memory and has 1,100,847,745 FLOPs.

Chapter 9.

Discussion

The methods introduced in this chapter have progressively enhanced the user's role in the annotation process, leading to a reduction in the amount of data that needs to be annotated for an effective monitoring. Initially, 110,422 bags of aggregate power signals were used as described in Chapter 5. However, with the application of weakly supervised transfer learning in Chapter 7, the number of bags was reduced to 7,635, which is a decrease of 93.08% from the initial set.

By further applying active learning guided by weak supervision (Chapter 8), the number of required annotated bags was reduced to 811 for four houses, marking a reduction of 89.37% compared to the weakly supervised transfer learning. It is worth noting that the annotation period for each of the houses can extend up to 36 days to achieve the reported performance.

In terms of performance, measured by the F_1-micro score, the initial network achieved a score of 47% when the dataset was mixed and tested on REFIT. The performance improved to 71% with the application of transfer learning. Finally, by selecting the most informative window in the active learning procedure, the average reported score increased to 78%.

In Figure 9.1, the scatter plot summarizes the number of annotated bags and the related performance for each methodology. "W" indicates the approach presented in Chapter 5 ([2]), "W-TL" indicates the approach presented in Chapter 7 ([3]), and "W-AL" indicates the approach proposed in Chapter 8 ([4]). The active learning based method, that collects both weak supervision and transfer learning, is the best both in terms of performance and in reducing the annotation effort.

In this part of the thesis, the proposed methodologies led NILM towards a user-centric view, where the developed strategies obtained good performance while considering the figure of the user and its role. The final result is a NILM approach that can perform correctly by using a bit more of one month of annotated data provided by the user.

Future works can include powerful tools to select the most significant bags. Explainability tools [112] can be involved to extract deep information from the unlabeled bags. Also, new re-training strategies can be considered to reduce

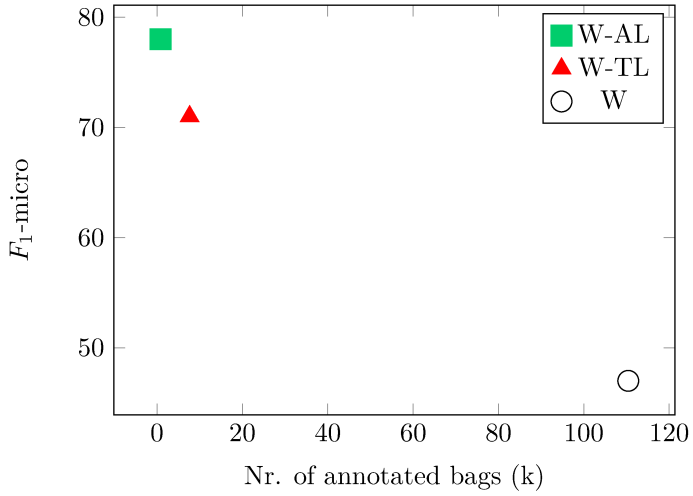


Figure 9.1.: Scatter plot for number of annotated bags vs performance expressed in terms of F_1-micro. "W": Chapter 5 ([2]), "W-TL": Chapter 7 ([3]), and "W-AL": Chapter 8 ([4]).

the training complexity of each iteration.

Part III.

Low-Complexity NILM Methods

The third contribution of this thesis, discussed in Chapter 2, will be presented in this part. NILM service should provide a real-time monitoring feedback, and it is fundamental to preserve the privacy of the users. These features can not be guaranteed by services installed on the cloud servers.

As it will be presented in Chapter 10, most of the current NILM state-of-the-art rely on powerful hardware due to the complexity of the approaches. Thus, to move the computation closer to the users, lower complexity approaches have to be developed.

A new distillation strategy based on weak labels is proposed in Chapter 11 to reduce the complexity in terms of number of parameters and floating point operations of the NILM approaches presented in the previous part of this thesis. The results demonstrate that reducing the architecture does not compromise performance, and in several cases performance are also improved.

Subsequently, based on a consistency analysis between the Teacher and Student networks, an explainability-based distillation strategy is proposed with effective performance in Chapter 12.

Then, the possibility to embed a new task (such as the monitoring of a new appliance) in a deployed network is considered to mimic a real scenario. In Chapter 13, a method is proposed that minimizes the number of parameters introduced. This method utilizes knowledge distillation to maintain the knowledge acquired from previous tasks. An advanced layer selection strategy is proposed, resulting in improved performance.

An overall discussion of this part is presented in Chapter 14. The results shown that both when reducing the complexity of a network or when introducing a new task with the minimum of complexity, the proposed approaches demonstrate to be solid.

Chapter 10.

Introduction

Supervised approaches require large datasets and typically utilize deep networks with millions of weights. These algorithms are developed using high-resource hardware. However, the spread of devices in human-daily life has offered the possibility to move some computations on the edge of the network, where these devices act. This means that the computation is closer to the user, preventing annoying service malfunctioning, preserving privacy, and reducing latency. But this means that the devices that should incorporate the service have limited computational resources. If the service is provided on the cloud, issues related to data transmission, privacy, and response latency can frequently occur. For instance, a Non-Intrusive Load Monitoring (NILM) method may be required in real-time to identify any irregularities in the usage patterns of one or more appliances. Additionally, the speed of data transmission can be affected by the resolution of the data being processed, making it susceptible to delays and interruptions. From a privacy perspective, initial insights into consumption patterns can indicate whether people are present in a building. A more detailed analysis could potentially uncover private aspects of the users' habits.

Since smart meter data are collected near the user, a local computation can improve NILM service performance. To address this, several strategies for complexity reduction have been proposed in the literature. Complexity reduction generally refers to the reduction of the number of trainable and non-trainable parameters that need to be saved after training and used in the prediction phase. It also refers to the reduction of the Floating Points Operations (FLOPs) required to produce the predictions.

Complexity reduction can be handled following two ways: the first consists in designing a large network and then apply model reduction strategies; the other consists in directly designing a lightweight architecture. In NILM, most of the works approached the problem by using a posteriori complexity reduction methodology.

Approaches for complexity reduction in NILM have primarily focused on neurons and filters pruning [113, 114, 115, 116], tensor decomposition [113],

and coefficient quantisation [117, 116]. In [113], filters are pruned based on their importance defined by L-norms and the change in the loss value caused by removing a specific filter and using the L1-norm. The same criteria have been adopted to prune the neurons. After pruning, the model is re-trained one or more times on a subset of training data, based on the adopted pruning strategy to recover possible decrease in performance. In [114], pruning techniques are used to reduce the complexity of a large Sequence-to-Point model [118] in a federated learning framework. The authors also addressed transfer learning by using unlabelled data from the target domain. Barber et al. [115] propose two ways to reduce the complexity of the Sequence-to-Point CNN network, using dropout and a smaller number of CNN filters and applying pruning on the learned weights. Particularly, four types of pruning approaches have been evaluated and the magnitude-based approach, implemented in the TensorFlow Model Optimization toolkit, was found to be the best compromise between reduction and accuracy of the model.

Federated learning for NILM has been introduced in [114, 54]. This learning model involves training on local devices using local data, and the model parameters are then sent to a central server to learn a global model. In FedNILM [114], the global model is fine-tuned on unseen and unlabeled measurements after pruning and network optimisation. In [54], a framework that merges federated learning and meta-learning is proposed, where a set of meta-learned models is locally trained using metering data from residential communities.

In [117], a post-training MobileNet compression is proposed, which reduces the model size and inference time using the TensorFlow Lite tool for quantisation, reducing the precision from 32-bit to 8-bit. Peng and colleagues [71] introduced a framework based on KD to achieve a Multi-Layer Perceptron network. A similar approach was used in [119], where KD was used to derive a CNN from an ensemble of convolutional networks, each with higher complexity. The task addressed here is multi-class single-label classification, meaning only one appliance is assumed to be active at each time instant. Conversely, in [120], the authors directly designed a lightweight CNN architecture suitable for deployment on edge devices.

Sykiotis and colleagues [116] presented an edge optimisation framework that incrementally applies coefficient quantisation and pruning to reduce the model's complexity until the specified performance and edge deployment requirements are met.

Most of the literature reviewed focuses on reducing complexity for power profile reconstruction [113, 114, 115, 116, 117]. However, only a few works consider the performance decline on unseen target domains and the associated transfer learning methods to mitigate it [114, 120]. This issue is crucial in practice as the data from the *source domain* used for network training often

differ statistically from the *target domain* data processed when the network is deployed in the final environment. Factors such as appliance types, measurement equipment, and building size can contribute to this statistical difference. Recent literature demonstrates that this mismatch between training and testing domains results in poor performance, necessitating transfer learning for satisfactory results [106, 54].

In [106], the authors transfer features extracted by the CNN layers of the Sequence-to-Point network across appliances and households in different regions and fine-tune the regression layer. In contrast, [54] combines federated learning and meta-learning, where a set of meta-learned models are locally trained using metering data from residential communities. Notably, neither [106] nor [54] propose a complexity reduction method and both deal with power profile reconstruction.

In the complexity reduction literature, only [120] has evaluated the method in a data domain different from the training one, and only [114] has addressed transfer learning. Both papers focus on power profile reconstruction, i.e., the regression task.

It is worth noting that Luan and colleagues [120] developed a lightweight architecture from scratch, eliminating the need for parameter reduction.

As referenced in Section 2.5, this chapter presents the third contribution of this thesis. It compiles innovative techniques for NILM, focusing on reducing complexity in transfer learning scenarios.

In Chapter 11, a new distillation method is introduced to decrease the complexity of the CRNN, both in terms of parameters and FLOPs. This method leverages weak supervision and fine-tuning to maintain or even enhance performance while reducing complexity.

An improvement of this work is proposed in Chapter 12, where the explainability technique is included in the distillation process to reduce the inconsistencies among the teacher and the student behaviour during the training, while reducing the complexity.

Following that, in Chapter 13, a continual learning strategy based on distillation is implemented for transfer learning. This strategy introduces a new task while using the fewest possible parameters.

Chapter 11.

Knowledge Distillation for Scalable Non-Intrusive Load Monitoring

This chapter offers an in-depth explanation of a novel Knowledge Distillation (KD) strategy for multi-label appliance NILM, as first introduced and published in [121].

This method employs KD and weak supervision to reduce the computational load a neural network for multi-label appliance classification. KD facilitates the transfer of knowledge from a larger 'teacher' network to a smaller 'student' network by training the latter with soft labels derived from the former [70, 122]. To enhance KD and ensure scalability of the proposed solution, both transfer learning and complexity reduction are necessary.

Transfer learning typically involves gathering new data directly from the target environment to fine-tune pre-trained models, which necessitates user involvement for data annotation. This process is simplified by weak supervision. Prior research has shown the effectiveness of weak labels in enhancing performance in both disaggregation [103] and multi-label appliance classification tasks [2, 3]. This study suggests using weak labels to simultaneously distil knowledge and reduce network complexity during transfer learning.

The method employs the CRNN presented in the above sections, since it has proven successful in a centralised NILM scenario [2, 3]. The experiments presented in the study feature several networks with reduced complexity that maintain the core components of the initial model, and explore the balance between accuracy and complexity.

11.1. Proposed Methodology

Figure 11.1 shows the proposed KD framework. Two learning phases, Pre-training and Fine-tuning, are performed on the Teacher network and one, Distillation, on the Student. The Teacher network is initially trained on a large dataset of active power measurements and the corresponding strong and weak

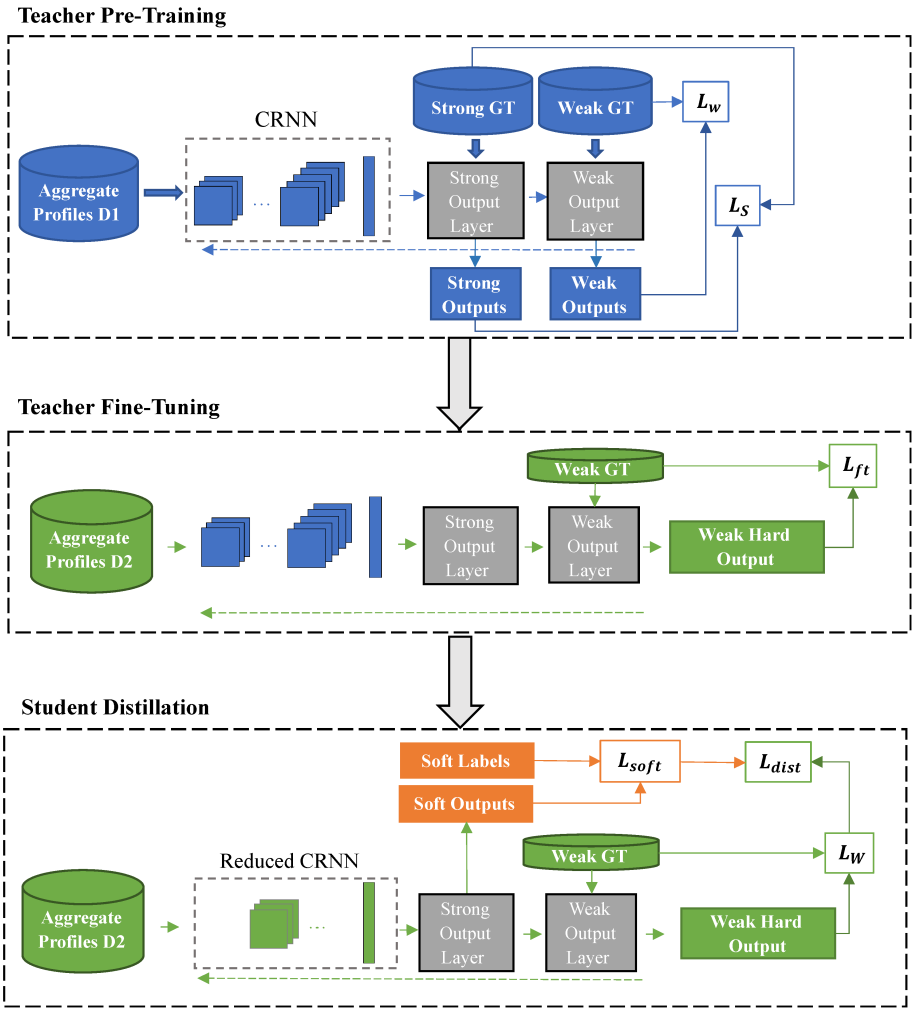


Figure 11.1.: Proposed Knowledge Distillation framework for NILM. "GT" stands for Ground Truth.

labels $\{\mathbf{y}_j, \mathbf{S}_j, \mathbf{w}_j\} \in D_1$. Then, the network is fine-tuned on a smaller set $\{\mathbf{y}_j, \mathbf{w}_j\} \in D_2$ without any strong labels.

To ensure the practicality of the proposed architecture, all learning phases are based on weak supervision [123]. That is, it is assumed that only the large teacher network has access to exact event labels (*strong* labels) in the pre-training phase, while the student network is created locally, at the target environment, with access to *weak* labels only. For example, the teacher can be trained using a large public source domain dataset, and fine-tuning is performed using easier-to-collect weak labels from the target domain (collected, e.g., periodically from the targeted house via an app).

The method is based on a weak supervised distillation approach in which the network takes as input a series of J disjointed windows of $y(t)$ of size L and produces as output a series of J disjointed windows of predictions for $\tilde{K}$ classes:

$$\hat{\mathbf{S}}_j = [\hat{\mathbf{s}}(jL), \hat{\mathbf{s}}(jL+1), \ldots, \hat{\mathbf{s}}(jL+L-1)] \in \mathbb{R}^{\tilde{K}\times L}, \tag{11.1}$$

where $\hat{\mathbf{s}}(t) \in \mathbb{R}^{\tilde{K}\times 1}$, contains predictions (on/off) for each of $\tilde{K}$ appliances of interest at time stamp t.

The distillation process is performed using the teacher-student strategy described in [70]. The following sections detail the teacher and the student training methodology. The final subsection is dedicated to the teacher architecture and the factors that influence the dimension of the network.

11.1.1. Teacher Learning

The Teacher model implements the function $g_\phi(\cdot)$ with parameters $\boldsymbol{\phi}$ and it is initially pre-trained using both strongly and weakly labelled data, i.e., the dataset D_1. The loss function is defined as:

$$L_{pt} = L_s + \lambda L_w, \tag{11.2}$$

where the two losses are the Binary Cross-Entropy (BCE) function calculated on the strong predictions and on the weak predictions, respectively, as Equation 5.7 and Equation 5.8.

Unlike previous works on distillation [70, 124], before being employed in the distillation process, the teacher network here is fine-tuned on a subset of data D_2 from the target environment using weak labels only. The fine-tuning loss

L_{ft} is formulated as the focal loss [125], with γ set to 0.2:

$$L_{ft}(\mathbf{w}_j, \hat{\mathbf{w}}_j) = -\frac{1}{\tilde{K}} \sum_{m=1}^{\tilde{K}} [w_m(1-\hat{w}_m)^{\gamma} \log(\hat{w}_m) + (1-w_m)\hat{w}_m^{\gamma} \log(1-\hat{w}_m)]\,. \quad (11.3)$$

Generally, positive and negative samples are highly unbalanced, as the latter are significantly more represented. Moreover, preliminary experiments on the validation set showed that the classification of negative samples is significantly less challenging, with specificity values around 0.99. This, motivated us to use the focal loss proposed in [125] instead of the binary cross-entropy loss. The focal loss focuses better on incorrect instances of the underrepresented class (positive samples in this case), while down-weighting the contribution of correctly classified samples related to the mostly represented class (negative samples in this case). In this way, the loss helps the Teacher in learning about the target domain data available before distillation, particularly when using the coarser information from weak labels. Experimentally, it is verified on the validation set that using the focal loss reduces the presence of false positive and negative predictions and increases the true positives depending on the appliance. All network layers have been fine-tuned since it has been verified on the validation set that better performance is obtained by re-training the entire network.

11.1.2. Student Knowledge Distillation

The Student model implements the function $f_{\alpha}(\cdot)$ with parameters $\boldsymbol{\alpha}$. The weakly labelled dataset D_2 exploited to fine-tune the teacher network has also been employed in the distillation process. Thus, the distillation loss function is defined as:

$$L_{dist} = \beta L_{soft}\left(\sigma\left(\frac{\mathbf{Z}_j^{st}}{T}\right), \sigma\left(\frac{\mathbf{Z}_j^{te}}{T}\right)\right) + (1-\beta)\theta(e) L_w(\hat{\mathbf{w}}_j^{st}, \mathbf{w}_j), \quad (11.4)$$

where L_{soft} is the BCE, as in (Equation 5.7), calculated on the soft outputs of the student $\tilde{\mathbf{S}}_j^{st} = \sigma(\mathbf{Z}_j^{st}/T)$ and the soft labels from the teacher $\tilde{\mathbf{S}}_j^{te} = \sigma(\mathbf{Z}_j^{te}/T)$ with σ being the sigmoid function, and $\mathbf{Z}_j^{st}$ and $\mathbf{Z}_j^{te}$ the logits from the Student and the Teacher, respectively. L_w is the BCE computed on the weak predictions $\hat{\mathbf{w}}_j^{st}$ of the student and $\mathbf{w}_j$ the weak ground-truth, as in (5.8). $\theta(e)$ is a dynamic weight that balances the magnitude of the two losses based on the following formula $\theta(e) = 10^{-G(e)}$, where $G(e)$ is obtained by $G(e) = \log_{10}(\mathcal{L}_w(e)) -$

Algorithm 3 Pseudo-code for the Student distillation process.

Require: Datasets D_1 and D_2, *Teacher* $g_{\boldsymbol{\phi}}(\cdot)$ pre-trained on D_1 and fine-tuned on D_2, *Student* $f_{\boldsymbol{\alpha}}(\cdot)$, $\theta(\cdot)$ function to balance losses magnitude.
  for e in *epochs* **do**
    for each minibatch B **do**
      $\tilde{\mathbf{S}}^{te}_{j\in B} \leftarrow g_{\phi}(\mathbf{y}_{j\in B})$;
      $\tilde{\mathbf{S}}^{st}_{j\in B}, \hat{\mathbf{w}}^{st}_{j\in B} \leftarrow f_{\alpha}(\mathbf{y}_{j\in B})$;
      $L_{dist} \leftarrow \beta L_{soft}(\tilde{\mathbf{S}}^{st}_{j\in B}, \tilde{\mathbf{S}}^{te}_{j\in B}) + (1-\beta)\theta(e)L_w(\hat{\mathbf{w}}^{st}_{j\in B}, \mathbf{w}_{j\in B})$;
  Update $\boldsymbol{\alpha}$ using Adam Optimiser to minimise L_{dist} loss.
    end for
  end for

$\log_{10}(\mathcal{L}_{soft}(e))$, e is the training epoch, and $\mathcal{L}_w(e)$ and $\mathcal{L}_w(e)$ are the total losses for epoch e. β balances the contribution of the teacher knowledge and the weak ground-truth to guide the training process. T is the temperature parameter used to soften teacher predictions [70]. β and T have been defined for each network architecture experimentally, based on the performance on the validation set. Figure 3 shows the pseudo-code for the Student distillation process.

11.1.3. Neural Network Architectures

The Teacher network is based on a CRNN, initially used in [2]. For the sake of clarity, the network structure is illustrated again here because in this section the focus is primarily on the architecture complexity. The network is composed of $H = 3$ convolutional blocks, each containing a convolutional layer with $F \cdot H$ filters ($F = 32$), with kernel size equal to $k_e = 5$, a batch normalisation layer, a Rectified Linear Unit (ReLU) activation and a dropout layer with probability equal to 0.1. The stride d is 1 and the padding modality is "same". The recurrent subpart is composed of a bidirectional Gated Recurrent Units (GRUs) layer, with 64 units (U). The final part of the network is composed of a dense layer with $\tilde{K}$ neurons followed by a sigmoid activation function that produces the appliances' state sample-by-sample. After the dense layer, the *linear softmax* pooling layer followed by a sigmoid activation layer, produces the weak prediction. Linear softmax pooling is chosen over other functions proposed in the literature as it is shown to reduce the incongruities between strong and weak labels leading to improved performance [60, 2].

The total number of trainable parameters for the convolutional subpart can be computed as:

$$N_{CNN} = \sum_{h=1}^{H} (k_e d \cdot F_{h-1} + 1)F_h + n_{BN}, \tag{11.5}$$

	Kettle		Microwave		Toaster		Washing Machine		Dishwasher		Washer Dryer	
Dataset	Mean Power	Duration	Mean Power	Duration	Mean Power	Duration	Mean Power	Duration	Mean Power	Duration	Mean Power	Duration
UK-DALE	1968	2.15	969	1.9	1437	3.38	512	85.8	802	106	504	85
REFIT	2066	2.4	961	2.5	1148	1.9	301	91.8	598	97	1060	30.4

Table 11.1.: Pre-training sets characteristics. Mean Power (expressed in Watt) refers to the mean power estimated in a complete activation while the Duration (expressed in minutes) refers to the length.

with $F_h = F \cdot h$, and $n_{BN} = 4F_h$ that represents the number of parameters associated to the batch normalisation (2 trainable plus 2 non-trainable). F_{h-1} is the number of feature maps in the input for the h-th layer while F_h is the number of feature maps in the output. When $h = 1$ the F_0 is the dimension of the input data. Thus, N_{CNN} mainly depends on the number of convolutional blocks. The recurrent subpart has a number of parameters N_{RNN} computed as [126]:

$$N_{RNN} = 2[3(U^2 + UF_H + 2U)], \tag{11.6}$$

where the last term depends on the used framework and is $2U$ for Keras and PyTorch. N_{RNN} depends on the number of recurrent units considered U, biases, and the input dimension F_H. Equations Equation 11.5 and Equation 11.6 indicate that the number of convolutional blocks and the number of recurrent units are the main factors that increase the total number of parameters and hence the overall complexity. In this work, several student architectures with reduced complexities are evaluated in the edge computing direction. The various student architectures are presented in Section 11.2.

11.2. Experimental Setup

11.2.1. Dataset

The appliances considered are Kettle (KE), Microwave (MW), Dishwasher (DW), Washing Machine (WM), Toaster (TOA), and Washer Dryer (WD) since they are present in most households and also present in most of the houses in both datasets. A subset of houses from REFIT (2, 4, 8, 9, 15) is used as a test set from which the set for fine-tuning D_2 the Teacher network has been extracted (30% of the total number of windows). The fine-tuning set is the same as that used for the distillation of the student.

To evaluate the approach in practical scenarios, two different pre-training sets D_1 for the Teacher are considered: (i) Houses 5, 6, 7, 10, 12, 13, 16, 17, 18 and 19 of REFIT; (ii) Houses 1, 3, 4, and 5 of UK-DALE. These houses are selected based on the availability of the six appliances of interest. The first scenario is to evaluate the method in more favourable conditions when the

pre-training domain is similar to the target data domain. The second allows us to evaluate the method performance when the pre-training and target data domains are statistically different [127]. The validation sets contain 20% of data from each training house. Input data are normalised using the mean and the standard deviation estimated on the pre-training sets.

11.2.2. Hyperparameters

The input sliding window dimension L in the Teacher model is the first hyperparameter that influences the distillation process. Table 11.1 shows the duration and average power values for all the appliances of interest. For long-activation appliances, the window size L is fixed to 4 hours and 15 minutes (2550 samples) as in [2] (Section 5), where this length is selected to ensure that a complete activation is contained within a window. Instead, a series of reduced window lengths is examined for short-activation appliances (around 2-4 minutes) after having analysed activations in both pre-training datasets. A total of four window lengths are identified, equally distributed from 55 minutes (540 samples) to 4 hours and 15 minutes (2550 samples). A minimum time interval of 55 minutes is chosen as it is appropriate for weak labels annotation. Thus, the selected window sizes are 55 minutes (540 samples), 2 hours and 2 minutes (1210 samples), 3 hours and 8 minutes (1880 samples) and 4 hours and 15 minutes (2550 samples). A smaller window for short-activation appliances makes weak labels more effective during the training phase, and multiple activations inside the same window can be accurately detected. Section 11.3.1 presents a comparison of the results obtained with windows of different lengths. From a practical point of view, using the one-hour windows for these appliances is a reasonable length for accurately assigning weak labels since users are less likely to remember appliances used within less than one-hour windows confidently.

The parameter β in Equation 11.4 has been varied in the range 0.3-0.9 with a step of 0.2, and T has been tested with values [0.5, 0.7, 0.9, 2]. β and T have been optimised for each student network based on the validation set that has also been used to find the best threshold to quantise the network predictions. The learning rate used is 0.002. The number of epochs has been set to 1000, and early stopping with patience equal to 30 epochs has been used to avoid over-fitting. The batch size is set to 64.

11.2.3. Architecture Complexity Evaluation

As introduced in Section 11.1.3, to reduce the Student architecture the main components of the Teacher network are maintained, and change the parameters of both convolutional and recurrent sub-parts. Firstly, the number of convolutional blocks is reduced and two structures, one with $H = 2$ and one with $H = 1$

Network	N_{CNN}	N_{RNN}
Teacher	52486	74496
Student	52486	74496
Student 2H-64U	10880	49920
Student 1H-64U	320	37632
Student 1H-32U	320	12672
Student 1H-16U	320	5219

Table 11.2.: Total number of parameters for convolutional and recurrent subparts are reported for the teacher network and each student architecture.

Network	Window size	Classes	FLOPs	Size (KB)
Teacher			56.4 M	551
Student			56.4 M	551
Student 2H-64U	540	3	11.9 M	284
Student 1H-64U	(55 min)		0.7 M	185
Student 1H-32U			0.5 M	85
Student 1H-16U			0.3 M	55
Teacher			267.8 M	552
Student			267.8 M	552
Student 2H-64U	2550	6	57.8 M	285
Student 1H-64U	(4h 15 min)		5.1 M	186
Student 1H-32U			3.1 M	87
Student 1H-16U			2.1 M	55

Table 11.3.: Number of FLOPs and sizes of teacher and student networks for different window lengths (in samples) and number of classes $K = 3$ and $K = 6$.

are considered. Then, $H = 1$ is fixed and start to decrease U by a factor of 2 to further reduce the architecture dimension and computational complexity. Table 11.2 reports the N_{CNN} and N_{RNN} for each architecture while Table 11.3 reports the number of Floating point Operations (FLOPs) and the dimension of the models to evaluate the reduction in terms of size and runtime [113]. The Student models are named with the number of H convolutional blocks and recurrent units U, e.g., Student 2H-64U denotes a student architecture with $H = 2$ convolutional blocks and $U = 64$ units. The model named Student has the same architecture of the Teacher. As shown in Table 11.3, the window dimension significantly affects the number of FLOPs.

11.2.4. Benchmark Methods

In the experiments, the proposed method is compared with two existing techniques in the literature that propose complexity reduction for NILM [113, 120]

and with [2]. None of the works presented in Chapter 10 proposes a complexity reduction approach for multi-label appliance classification. Therefore, [113, 120] were adapted for this task. EdgeNILM [113] uses pruning and tensor decomposition applied to [118], and in the experiments the source code made available by the authors is used to ensure reproducibility. To adapt the network to multi-label appliance classification, the last layer of the Sequence-to-Point CNN is modified with a sigmoid function to produce the state probability and used the BCE loss function during training. As in [113], a separate network is trained for each appliance and applied the 60% iterative pruning complexity reduction method because in [113] it produced the average lowest disaggregation error. A window size of 99 samples was adopted for EdgeNILM for all the appliances, based on the results presented in [113].

The LightweightCNN proposed in [120] is based on a model design approach, and it consists of only two convolutional layers and one dense layer. The lightweight network was implemented and trained within the same framework of EdgeNILM for a fair comparison, using a window size of 199 samples [120]. As with EdgeNILM, for this approach, a separate network for each appliance is trained.

Finally, the proposed method is compared with the initial work proposed in [2], where the CRNN structure was trained with weakly labelled data. This method is identified as WL-NILM. In this way, the effectiveness and novelty of method is demonstrated in terms of complexity-performance improvement also when compared with an approach that uses a CRNN and weak labels during training.

The same post-processing applied for proposed method was applied to the raw predictions of benchmark methods, using a threshold for each network optimised on the validation set.

11.2.5. Evaluation Metrics

Two metrics have been used to evaluate the proposed approach. The first is the F_1-score (F_1) and the related micro-average already defined in Equation 5.9 and Equation 5.10.

The load estimation is evaluated using the TECA Equation 5.11. The average power consumed by each appliance has been assigned based on the average power consumed by the appliances in the training set.

11.3. Results and Discussion

In this section, the window length impact on the Teacher performance is firstly illustrated. Then, the results obtained by the reduced Student networks are

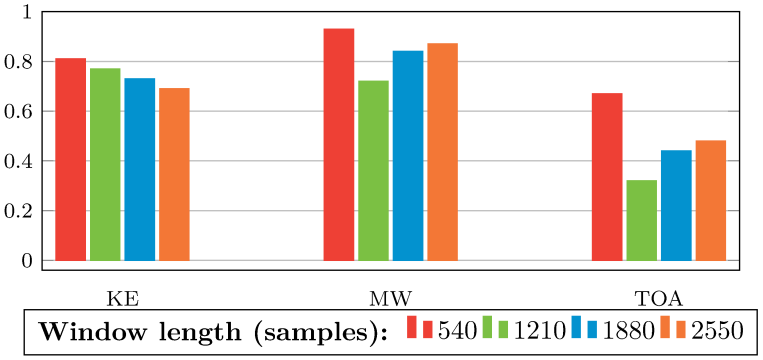


Figure 11.2.: Window analysis based on the test set: F_1-scores when the Teacher is pre-trained with REFIT.

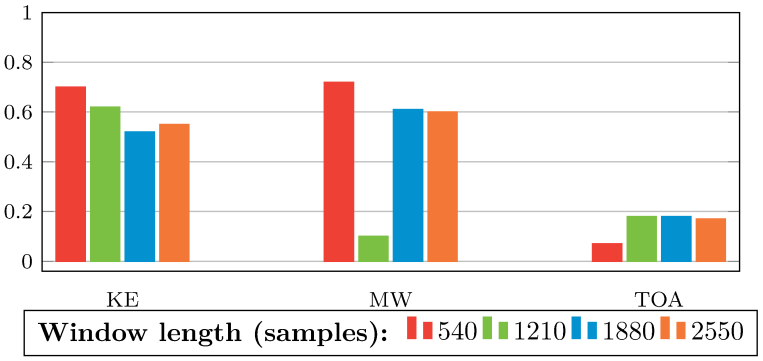


Figure 11.3.: Window analysis based on the test set: F_1-scores when the Teacher is pre-trained with UK-DALE.

provided. Lastly, the comparison with benchmark methods is discussed.

11.3.1. Window Length Impact on Teacher Performance

Teacher performance for Kettle, Microwave, and Toaster are presented to evaluate the best window length for classifying short-activation appliances. In this way, the hypothesis that using a window shorter than the one used in [2] leads to improved performance is validated. Figures 11.2 and 11.3 report the Teacher performance after fine-tuning on the target data for different window lengths for aforementioned appliances when pre-training is performed on REFIT and UK-DALE, respectively. Although only the results on the test set are reported, the performance on the validation set reflects the performance on the test set.

Both figures show that the reduced length window of 540 samples enables more effective detection of the appliances' states. This is confirmed for both pre-training set conditions and all the appliances except for the Toaster. The

Toaster's performance is affected by the statistical differences in power and duration between the activations in the pre-training and the test set, leading to a small drop in an already poor performance. For Microwave, the difference in duration between activations from different domains is reduced when the network focuses on a shorter time window.

	Teacher			Student			Student 2H-64U			Student 1H-64U			Student 1H-32U			Student 1H-16U		
Appliance	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	PR	R	F_1	P	R	F_1
KE	0.89	0.74	0.81	0.88	0.75	0.81	0.89	0.66	0.76	0.89	0.74	**0.81**	0.88	0.74	0.80	0.86	0.81	**0.83**
MW	0.88	0.98	0.93	0.85	0.98	0.91	0.82	0.98	0.90	0.83	0.98	0.90	0.85	0.98	0.91	0.74	0.93	0.82
TOA	0.52	0.94	0.67	0.77	0.74	**0.76**	0.76	0.73	**0.74**	0.77	0.71	**0.74**	0.79	0.75	**0.77**	0.03	0.0	0.0
WM	0.57	0.91	0.70	0.56	0.95	**0.71**	0.62	0.92	**0.74**	0.61	0.92	**0.74**	0.58	0.94	**0.72**	0.58	0.93	**0.71**
DW	0.35	0.97	0.51	0.36	0.98	**0.52**	0.38	0.97	**0.55**	0.38	0.97	**0.55**	0.39	0.98	**0.56**	0.42	0.93	**0.58**
WD	0.93	0.52	0.67	0.97	0.40	0.57	0.98	0.38	0.55	0.97	0.44	0.60	0.91	0.61	**0.73**	0.98	0.34	0.51
AVG.	0.69	0.84	0.71	0.73	0.80	**0.73**	0.74	0.77	**0.71**	0.72	0.83	**0.72**	0.73	0.83	**0.75**	0.60	0.66	0.57

Table 11.4.: Performance comparison between the Teacher (D_1=REFIT) and the Student networks. Improved performance are in bold and underlined while equal performance are in bold for the reduced Student architectures.

	Teacher			Student			Student 2H-64U			Student 1H-64U			Student 1H-32U			Student 1H-16U		
Appliance	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	PR	R	F_1	P	R	F_1
KE	0.60	0.84	0.70	0.61	0.86	**0.71**	0.56	0.82	0.67	0.58	0.90	**0.71**	0.62	0.87	**0.73**	0.59	0.88	**0.70**
MW	0.57	0.99	0.72	0.53	0.99	0.69	0.50	0.99	0.66	0.62	0.97	**0.75**	0.61	0.95	**0.75**	0.68	0.97	**0.80**
TOA	0.22	0.04	0.07	0.26	0.04	0.07	0.31	0.03	0.05	0.06	0.01	0.02	0.07	0.0	0.0	0.25	0.04	**0.07**
WM	0.56	0.69	0.62	0.57	0.74	**0.65**	0.62	0.67	**0.65**	0.70	0.40	0.51	0.52	0.78	**0.63**	0.69	0.35	0.46
DW	0.49	0.84	0.62	0.50	0.87	**0.63**	0.48	0.88	**0.62**	0.38	0.92	0.54	0.39	0.93	0.55	0.39	0.92	0.54
WD	0.79	0.77	0.78	0.76	0.79	0.78	0.75	0.79	0.77	0.79	0.76	**0.78**	0.75	0.80	**0.78**	0.73	0.82	0.77
AVG.	0.54	0.70	0.59	0.54	0.72	0.59	0.57	0.65	0.57	0.52	0.66	0.55	0.49	0.72	0.57	0.55	0.66	0.56

Table 11.5.: Performance comparison between the Teacher (D_1=UK-DALE) and the Student networks. Improved performance are in bold and underlined while equal performance are in bold for the reduced Student architectures.

Appliances	Teacher	Student	Student 2H-64U	Student 1H-64U	Student 1H-32U	Student 1H-16U
KE, MW, TOA	0.827	**0.832**	0.820	**0.828**	**0.827**	0.822
WM, DW, WD	0.648	**0.657**	**0.656**	**0.680**	**0.740**	0.644

Table 11.6.: Performance comparison between the Teacher (D_1=REFIT) and the reduced Student networks in terms of TECA. Improved and equal performance are reported in bold for each Student architecture.

Appliances	Teacher	Student	Student 2H-64U	Student 1H-64U	Student 1H-32U	Student 1H-16U
KE, MW, TOA	0.627	**0.631**	0.608	0.624	**0.663**	**0.642**
WM, DW, WD	0.725	0.719	0.713	0.688	0.674	0.669

Table 11.7.: Performance comparison between the Teacher (D_1=UK-DALE) and the reduced Student networks in terms of TECA. Improved and equal performance are reported in bold for each Student architecture.

11.3.2. Student Distillation Results

Table 11.4, Table 11.5 present the results obtained with different student architectures, compared to the Teacher performance for all the $K = 6$ appliances. When using UK-DALE for pre-training, the Student network shows similar performance to the Teacher network with slight improvement for Kettle, Dishwasher, and Washing Machine. Similarly, when the Teacher is pre-trained with REFIT, the results are either improved or similar for Kettle, Toaster, Washing Machine and Dishwasher. A significant drop in performance is observed only for Washer Dryer due to low Recall. This is because Washer Dryer activations in the test set are longer than the activations in REFIT pre-training set (approximately 82 minutes vs 30 minutes). These statistical differences cause the network to miss or underestimate more activations, producing more false negatives. When the Student architecture is reduced, differences between domains become more critical because the network loses the last convolutional block related to higher-level features. In fact, for the Student 2H-64U network, N_{CNN} reduces by 79% and N_{RNN} by 33% compared to the Teacher, while the F_1-score reduces only by 3.4%, on average, due to a decrease in Recall not compensated by the slight increase in Precision. In contrast, for the same Student 2H-64U architecture distilled by the Teacher pre-trained on REFIT, the performance improves for the Toaster, Washing Machine, and Dishwasher, and remains stable for other appliances, except for Washer Dryer due to low Recall. In the smaller Student 1H-64U network, N_{CNN} reduces by 99% and N_{RNN} by 49%, while the F_1-score decreases by 6.8% due to both Recall and Precision drop after the distillation from the Teacher pre-trained on UK-DALE. This important reduction of high-level features affects the performance, particularly for Toaster, Dishwasher, and Washing Machine. Nonetheless, Kettle

and Microwave are more accurately classified while Washer Dryer maintains stable performance. When the Teacher is pre-trained on REFIT, the F_1-score of Student 1H-64U improves by 1.4% on average compared to the Teacher, with stable performance for Kettle, an improvement for Toaster, Dishwasher, and Washing Machine, with an exception for Microwave and Washer Dryer that slightly decrease. In this case, the network produces fewer false activations compared to the Teacher network, as confirmed by the higher Precision.

The Student 1H-32U (N_{RNN} reduced by 83%) represents a good compromise between complexity reduction and performance. This architecture improves Teacher performance in both pre-training scenarios. This behaviour shows that this architecture helps to improve Student generalisation ability independently of the pre-training set characteristics.

For the Student 1H-16U (N_{RNN} reduced by 93%), the F_1-score decreases for appliances with longer activations (26% for Washing Machine, 13% for Dishwasher, and 1% for Washer Dryer), while Kettle, Microwave, and Toaster have increased performance, compared to the Teacher pre-trained on UK-DALE. Particularly, activations of Washing Machine are not well detected while more false activations have been produced for Dishwasher and Washer Dryer. The performance indicates that the number of recurring units may be too small to learn patterns of household appliances with longer activation, when the domains are very different. In fact, the Student 1H-16U distilled from the Teacher pre-trained on REFIT has good performance for Kettle and longer activations appliances, like Dishwasher and Washing Machine, while for Microwave and Washer Dryer, the performance is reduced by 12% and 24%, respectively. It has to be noted that each reduced Student architecture reports an improvement for the Washing Machine and Dishwasher, suggesting that when domains are similar the classification of these appliances is positively influenced by complexity reduction. Conversely, Microwave performance slightly decreases compared to the Teacher for each student configuration due to the higher presence of false activations. Washer Dryer and Kettle are more dependent on the structure of the Student, while Toaster seems to be independent except for the Student 1H-16U where performance falls to 0%. The same holds when the reduced Student networks are distilled from a Teacher pre-trained on UK-DALE, mainly due to Teacher capability.

Due to the differences between the domains and loads characteristics, all the appliances are more influenced by the Student structure, and performance varies for each architecture. Nonetheless, Student 1H-32U performs better than the other structures, with the smallest performance degradation (3% for UK-DALE pre-training) and highest performance improvement (6% for REFIT pre-training) with a reduction of 10x in number of parameters, coherently in both pre-training scenarios. This outcome can be motivated by a good balance

Table 11.8.: Results in terms of F_1-score of the proposed approach and benchmark methods trained with D_1=REFIT and tested on REFIT. Best results are reported in bold.

	Appliance						
Model	**KE**	**MW**	**WM**	**DW**	**TOA**	**WD**	**Average**
EdgeNILM Unpruned [113]	0.81	0.41	0.19	0.31	0.21	0.41	0.39
EdgeNILM Pruned 60% [113]	**0.82**	0.29	0.19	0.31	0.11	0.51	0.37
LightweightCNN [120]	0.74	0.65	0.34	0.62	0.11	0.32	0.46
WL-NILM [2]	0.74	0.71	0.54	0.43	0.25	0.02	0.45
Teacher	0.81	**0.93**	0.67	0.70	0.51	0.67	0.71
Student 1H-32U	0.80	0.91	**0.77**	**0.72**	**0.56**	**0.73**	**0.75**

between the number of convolutional blocks, that extract only local features, and the number of recurrent units that take the features as input. The results in Table 11.6 and Table 11.7 show a comparison between the network structures in terms of TECA, where long- and short-duration appliances are considered separately. For appliances with shorter activations, when the Teacher is pre-trained with UK-DALE, there is a decrease in energy estimation of 3% for Student 2H-64U and of 0.5% for Student 1H-64U. For other architectures, the energy is estimated better than the Teacher, or the performance is similar. On the other hand, the TECA for long-activation appliances progressively reduces with the Student architecture reduction due to the slight progressive degradation of either Precision and Recall, especially for Student 1H-16U, for which the activations are underestimated for Washing Machine and overestimated for the Dishwasher. This result shows the variability of performance depending on the Student structure for long-activation appliances influenced by the appliances' characteristics that are very different between the two domains in terms of power values and duration. With shallow architectures, transfer learning process does not sufficiently improve the model. When the pre-training is performed with REFIT, the TECA is either similar or improved for long-activation appliances, because of data statistical similarity between the source and target environment in this case, except for Washer Dryer. The same holds for short-activation appliances, with a decrease of only 0.8%.

In summary, in the same domain the proposed method reduces the complexity and improves the performance. When domains are different, the performance is similar but the complexity is significantly reduced. The proposed method reduces the complexity and maintain acceptable performance, reducing, in the best case, 86x the FLOPs, and 10x the number of parameters.

Table 11.9.: Results in terms of F_1-score of the proposed approach and benchmark methods trained with D_1=UK-DALE and tested on REFIT. Best results are reported in bold.

	Appliance						
Model	**KE**	**MW**	**WM**	**DW**	**TOA**	**WD**	**Average**
EdgeNILM Unpruned [113]	0.64	0.01	0.43	0.19	0.02	0.23	0.25
EdgeNILM Pruned 60% [113]	0.68	0.03	-	0.07	0.02	-	0.13
LightweightCNN [120]	**0.75**	0.33	0.51	0.53	0.06	0.42	0.43
WL-NILM [2]	0.73	0.07	0.10	0.44	0.04	0.14	0.26
Teacher	0.70	0.72	0.62	**0.62**	**0.07**	0.68	**0.59**
Student 1H-32U	0.73	**0.75**	**0.63**	0.55	0.0	**0.78**	0.57

11.3.3. Comparison with Benchmark Methods

Table 11.8 and Table 11.9 report the results of the proposed method compared to benchmark approaches. For EdgeNILM the results of the model before pruning are reported, and the Teacher performance are included to facilitate evaluation and comparison the methods.

In both pre-training domains, the proposed approach outperforms the benchmark methods on average and for almost all the appliances. The Kettle is the only exception, where the LightweightCNN and pruned EdgeNILM achieve slightly better F_1-score, respectively, when trained using the UK-DALE and REFIT datasets.

Pruning improved the performance of EdgeNILM on the Kettle and Washer Dryer appliances when pre-trained with D_1=REFIT, but the performance of the other appliances remained relatively stable. On average, the performance of EdgeNILM Pruned 60% are worse than EdgeNILM Unpruned.

LightweightCNN demonstrated better performance on average for all the appliances compared to EdgeNILM, in particular for Microwave, Washing Machine, and Dishwasher. Instead, compared to WL-NILM, LightweightCNN

Table 11.10.: Model Size (MB) and FLOPS (M) for the benchmark methods and the proposed approach. The model size and the number of FLOPs are calculated on all the networks used to classify $K = 6$ appliances.

Model	Size (MB)	FLOPS (M)
EdgeNILM Unpruned [113]	82.92	38.54
EdgeNILM Pruned 60% [113]	13.38	6.28
LightweightCNN [120]	12.78	6.12
WL-NILM [2]	0.55	267.8
Teacher	1.10	324.2
Student 1H-32U	**0.172**	**3.6**

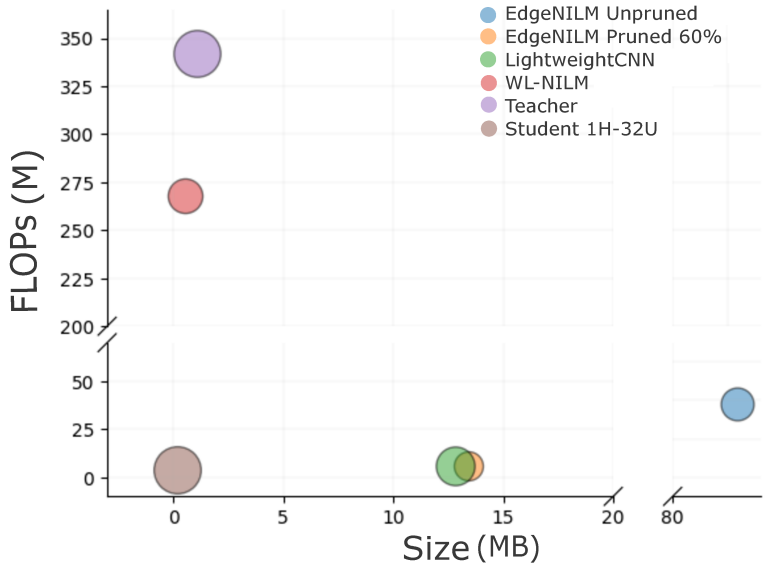


Figure 11.4.: Complexity-Performance comparison among benchmark methods and the proposed method. For each approach the dimension of the circle is proportional to the mean F_1-score of both D_1 scenarios. FLOPs are expressed in Millions (M) and Size is expressed in megabytes (MB).

is less effective for all the appliances except the Washer Dryer. Nonetheless, the proposed Student network has a higher F_1-score compared to EdgeNILM Pruned 60%, LightweightCNN and WL-NILM with an absolute increment of 0.38, 0.29 and 0.30, respectively.

When pre-trained with D_1=UK-DALE, the differences among domains has a greater impact on EdgeNILM and WL-NILM, which show low performance for all the appliances except the Kettle. In particular, for EdgeNILM Pruned 60%, Washing Machine and Washer Dryer are not reported because the model was not able to learn with a high pruning percentage. Except for Kettle and Dishwasher, WL-NILM produces poor results like EdgeNILM for all other appliance. For LightweightCNN, performance only slightly decreases with respect to the other pre-training domain. Also in this domain, proposed approach is more effective on average, with an absolute increment of 0.44, 0.14, and 0.31, on EdgeNILM Pruned 60%, LightweightCNN, and WL-NILM respectively. Particularly for EdgeNILM and WL-NILM, the absence of transfer learning in the complexity reduction process largely affects the performance on a different domain.

Table 11.10 reports the model size and the FLOPs for each approach, considering the total number of networks involved in the classification of $K = 6$ ap-

pliances. It is worth noting that EdgeNILM pruned 60% and LightweightCNN have almost the same number of FLOPs and model size, although the latter approach has shown better performance. Instead, WL-NILM has a higher number of FLOPs compared to EdgeNILM and LightweightCNN, with performance that varies depending on the pre-training domain. Nonetheless, the proposed Student has a number of FLOPs 1.74, 1.7 and 74.4 times smaller than EdgeNILM Pruned 60%, LightweightCNN and WL-NILM respectively, despite using a larger or equal window dimension than the benchmark methods, a parameter that affects the number of FLOPs (Table 11.3). Note that the model size of the proposed approach is 78, 74, and 3 times smaller than the benchmarks, while reporting superior performance. Considering both, the complexity of the architecture and the performance, the proposed Student network is more efficient and effective than the benchmark methods in appliance classification. Figure 11.4 shows a complexity-performance comparison among the benchmarks and the proposed method, where the circle dimension is proportional to the mean F_1-score computed on both D_1 pre-training datasets. WL-NILM and EdgeNILM Unpruned are on the opposite side of the plane, remarking that the difference in terms of FLOPs is mainly related to the window dimension of WL-NILM that is around 25 times wider. On the other hand, although proposed Student network has the same window dimension, the number of FLOPs is largely reduced compared to WL-NILM while producing better predictions. Considering the model size, the same can be highlighted compared to the other approaches, that present larger sizes with lower performance.

Chapter 12.

Improving Knowledge Distillation through Explainability Guided Learning

In other application domains, KD has demonstrated effective results in maintaining the performance of the Teacher network, while facilitating scalability [128] and preservation of privacy [129]. An important issue that has received considerable critical attention in deep learning and NILM community is algorithmic transparency [130, 131, 132]. Lack of interpretability brought by the algorithmic complexity of DNN models has caused many to regard them as “black-box” algorithms, leading to concerns raised by the scientific community [133], as well as legislative bodies [134]. This problem has been a focus of field of explainable AI (XAI), aimed to derive methods for creation of more trustworthy deep learning systems by providing human-understandable explanations of DNN outputs. Previous studies in this area have sought to propose techniques for generating visual explanations that highlight the features of the input which are the most influential for the prediction of a model. A considerable volume of literature suggests that such approaches can lead to more trustworthy machine learning systems [130, 132, 135, 136, 137]. However, despite the apparent benefits of introducing XAI in DNN-based NILM systems, most studies in KD NILM have only focused on domain adaptation and architecture reduction [138, 139, 121], and little is understood about the mechanism behind the transfer of knowledge from the Teacher to the Student model. Importantly, the relationship between the explanations of the Teacher model outputs and how they relate to explanations of the Student model decisions has not received any attention in the NILM community.

In this chapter, a methodology that establishes a link between KD and XAI approaches for NILM is proposed. A KD framework is used to train less complex networks (Students) for each appliance starting from a more complex network (Teacher) trained on a large quantity of samples from different domains. The Teacher network is a multi-label classifier used to distill the knowledge

into a binary Student classifier model. By exploiting existing XAI tools, visual explanations of outputs generated by the components of the KD system are derived, with the aim of understanding the distillation mechanism. This information is used to identify the main type of inconsistency w.r.t. transfer of explanation knowledge. Finally, a method for improvement of predictive performance of KD NILM algorithms is proposed, by guiding the distillation process towards correct transfer of explanation knowledge. This work has been presented and published at the IEEE International Conference on Acoustics, Speech and Signal Processing in 2023 [112].

12.1. Proposed Methodology

In this section, the KD framework for DNN architecture reduction is illustrated. Then, the proposed technique to generate the output-related explainability maps and explainability guided learning to enhance the training process of a distillation model. The method aims to explain why the Teacher predictions are produced and exploit this knowledge to improve the training process. Then, interpretability is included in the training loop.

12.2. Knowledge Distillation

The distillation framework mainly refers to Section 11.1. The architectural reduction compared to the Teacher is achieved by reducing the number of convolutional blocks and gated recurrent units in the Student model, leading to a 6-fold reduction in the number of trainable parameters. The architectures of Teacher and Student networks considered are shown in Table 12.1. The Teacher network is pre-trained on a large set of aggregate smart meter load profiles and then fine-tuned on a smaller set of aggregate signals. The pre-training set is annotated with sample-by-sample labels (*strong* labels) and bag-level labels (*weak* labels). The networks take as input a series of D disjointed aggregate windows with dimension L and produce as output two levels of predictions, a series of D sample-by-sample state predictions $\hat{x}_s \in R^{1\times L}$ at the *strong* level and a series of D window predictions $\hat{w}_s \in R^{1\times 1}$ at the *weak* level. Both levels are shown in Table 12.1. The pre-training loss at the Teacher network is formulated as $L_{pt} = L_s + \lambda L_w$, with L_s and L_w being 5.7 and 5.8 for strong and weak predictions, respectively. Then, the Teacher network is fine-tuned on a set of mains, annotated only with weak labels and the same set is also used during the distillation process for the Student network training. Fine-tuning is performed by re-training the Teacher network with the loss function defined as $L_{ft} = L_w$. The distillation loss compares soft Teacher with soft

Model	Layer	Activation	Filters	Kernel	Units
Teacher	Convolutional Block 1	ReLu	32	5	-
	Convolutional Block 2	ReLu	64	5	-
	Convolutional Block 3	ReLu	128	5	-
	Bidirectional GRUs		-	-	64
	Fully Connected (*strong* level)	Sigmoid	-	-	5
	Linear Softmax Pooling	-	-	-	5
	Activation (*weak* level)	Sigmoid	-	-	-
Student	Convolutional Block 1	ReLu	32	5	-
	Bidirectional GRUs		-	-	32
	Fully Connected (*strong* level)	Sigmoid	-	-	5
	Linear Softmax Pooling	-	-	-	5
	Activation (*weak* level)	Sigmoid	-	-	-

Table 12.1.: Architecture of Teacher and Student models.

Student predictions and weak level predictions with weak ground-truth, and it is formulated as:

$$L_{KD} = \beta \cdot L_{soft}\left(\sigma\left(\frac{\hat{x}_s}{T}\right), \sigma\left(\frac{\hat{x}_t}{T}\right)\right) + (1-\beta)\cdot\theta(e)\cdot L_w(\hat{w}_s, w), \qquad (12.1)$$

with $\sigma(\hat{x}_s/T)$ being soft predictions of the Student and $\sigma(\hat{x}_t/T)$ soft labels from the Teacher, and σ being the sigmoid function. T is the temperature parameter used to soften Teacher predictions [70]. $\theta(e)$ is a dynamic weight that balances the magnitude of the two losses based on the formula $\theta(e) = 10^{-G(e)}$ where $G(e)$ is obtained by $G(e) = \log_{10}(L_w(e)) - \log_{10}(L_{soft}(e))$ and index e is the training epoch. Parameter β balances the contribution of the Teacher knowledge and the weak ground-truth. At the end of the distillation process, Student predictions are quantized to obtain the state of the appliance, by applying a threshold selected based on the validation set.

12.3. Feature Importance Map Generation

As the need for explainability is becoming an increasingly important step for integration of AI systems, there has been a strong push towards development of practical tools that facilitate better understanding of complex, "black-box" algorithms. In order to incorporate XAI in the NILM KD framework, the attetion is placed on GradCAM, one of the most cited explainability methods [140]. GradCAM aims to solve the problem of assigning importance values to the input features of a DNN algorithm.

Given an input x to a DNN model, and a target concept c, the goal is to map the relevance of each input feature to the target concept, where the target concept can be represented as a class of interest in the case of classification tasks. GradCAM operates by computing the gradient w.r.t the final convolutional layer of a CNN network [140]. In order to generate an explanation map

$h^c \in R^{W \times H}$ of width W and height H for a target concept c, the gradient of the output for the target concept y^c w.r.t the kth feature map activations A^k of the last convolutional layer is computed, i.e., $\frac{\partial y^c}{\partial A^k}$. Next, a global average pooling operation is applied over the height and width dimensions (indexed by i and j, respectively) on the computed gradients, to obtain neuron importance weights [140]:

$$\omega_k^c = \frac{1}{W \times H} \sum_i \sum_j \frac{\partial y^c}{\partial A_{i,j}^k}. \tag{12.2}$$

The generated weights represent the importance of feature map k for the target concept c. In order to compute the explanation map h^c, weighted combination of feature map activations, followed by ReLU function, is performed [140]:

$$h^c = ReLU\left(\sum_k \omega_k^c A^k\right). \tag{12.3}$$

Note that ReLU operation ensures that only features with a positive influence on the target concept are considered.

12.4. Explainability Guided Learning

KD minimizes the divergence between the probability distributions of the Teacher and Student models [70], with the aim of aligning the logits produced by the Student with those of the Teacher. This process achieves effective transfer of knowledge by conditioning the Student model to mimic the outputs of the Teacher. However, it can observed that KD might not always be successful in transferring the explainable knowledge of the Teacher. In particular, the main erroneous case of inconsistency in the explanation knowledge transfer, that is, given identical inputs, Teacher and Student networks produce dissimilar output explanations for a given class. This phenomenon is illustrated with an example in Fig. 12.1 a)-b) in the form of a heatmap, where the highest values correspond to input features most important for the predictive output of the Washing Machine class. The distillation process has been unsuccessful in transferring the magnitudes of most relevant importance values to the Student, possibly causing the occurrence of a false positive prediction. A reduction of such inconsistencies might be a crucial step in the optimization of the distillation process, leading to a more stable predictive performance.

To prevent inconsistencies in the transfer of explainable knowledge, a learning technique for improvement of knowledge distillation is derived, focusing on dissimilarities between the Teacher and Student explanations. The distillation process is conditioned to transfer the Teacher behaviour both in terms

Appliance	**Scenario**	γ	μ
Washing Machine	UK-DALE	0.50	*weak*
	REFIT	0.30	*strong*
Dishwasher	UK-DALE	0.85	*strong*
	REFIT	0.70	*weak*
Washer-Dryer	UK-DALE	0.60	*weak*
	REFIT	0.30	*weak*
Kettle	UK-DALE	0.30	*weak*
	REFIT	0.70	*weak*
Microwave	UK-DALE	0.70	*weak*
	REFIT	0.5	*strong*

Table 12.2.: Training hyperparameters used for training of Student models for each of the two domain adaptation scenarios.

of output predictions and output explanations. This mode of learning, hereinafter *explainability guided learning*, is achieved through a new distillation loss function, modified to guide the learning process towards the resolution of explanation inconsistencies. As explanation heatmaps are represented in vector form, the inconsistency between two explanations are quantified through a loss function based on a measure of cosine similarity, defined as:

$$L^{\mu}_{xai}(a,b) = -\frac{ab}{\|a\|\|b\|} = -\frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n}(a_i)^2}\sqrt{\sum_{i=1}^{n}(b_i)^2}}, \tag{12.4}$$

where a and b represent two generated explanations, while μ represents the output type to be compared (weak or strong). It is expected that two similar vectors will have a similar angle between them, leading to the conclusion that the similarity of two vectors increases as the value of their cosine angle increases. To this end, in order to promote the minimization of the loss function, the sign of the generated cosine similarity measure is inverted.

To alleviate inconsistencies w.r.t transfer of explainable knowledge in KD, the KD loss function is modified by including the cosine similarity-based loss between the explanations produced by the Teacher and the Student networks. Thus, the explainability guided knowledge distillation loss function can be defined as:

$$L_{XGKD} = L_{KD} + \gamma \cdot L^{\mu}_{xai}(h_t, h_s), \tag{12.5}$$

where h_t and h_s represent explanations generated by Teacher and Student networks, respectively, while γ represents a parameter that adjusts the impact of the cosine similarity loss component L^{μ}_{xai}.

12.5. Experimental setup

12.5.1. Datasets

To validate proposed proposed approach, real-world UK-DALE [24] and REFIT [25] datasets are used. To evaluate the success of approach in performing domain adaptation, two different scenarios are used to pre-train the Teacher network, where training data are taken from 1) UK-DALE houses 1, 3, 4, and 5 (UK-DALE-to-REFIT scenario) and 2) REFIT houses 5, 6, 7, 10, 12, 13, 16, 17, 18 and 19 (REFIT-to-REFIT scenario). The UK-DALE-to-REFIT scenario is used to evaluate the performance of the proposed method when pre-training and target environment domains are different, while the REFIT-to-REFIT scenario aims to evaluate the performance of the method when the pre-training domain is similar to the target environment signal domain. The validation set for each scenario is extracted from the pre-training set, as well as the mean and standard deviation values used to normalize the input signals.

12.5.2. Training Procedure

The approach is evaluated on five appliances (Washing Machine (WM), Dishwasher (DW), Washer-Dryer (WD), Kettle (KT), and Microwave (MW)), across two domain adaptation scenarios (UK-DALE-to-REFIT and REFIT-to-REFIT). Teacher is trained to perform multi-label classification of an input signal. As part of distillation framework, the Student model is designed as a binary classifier with reduced architecture compared to the Teacher so that explainability guided learning can be focused on explanations for one appliance/class at a time. Moreover, the model can be used without re-training, even if some of the five appliances of interest are not present in the target house. Firstly, the knowledge distillation is performed without explainability guided learning, using L_{KD} loss defined in Eq. (12.1), to create baseline Student models for each appliance in the two domain adaptation scenarios. Then, the same process is repeated with explainability guided learning with a loss function defined in Eq. (12.5). As each appliance model is sensitive to the choice of μ and γ, the chosen hyperparameters are reported in Table 12.2. Hyperparameters and thresholds to quantize the predictions have been selected for each model such that they maximize the performance on the validation set. The input window dimension is $L = 2550$ which corresponds to 4h and 15min of measurements. The batch size is set to 64. Adam optimizer is used with a learning rate of 0.002, and a number of epochs is set to 1000.

12.6. Results

Standard classification metrics: Recall, Precision, and F1-score, are used for evaluation as in Section 5.2.3.

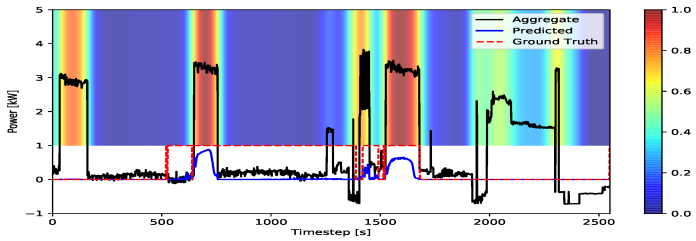
Firstly, in Table 12.3, results are presented for the case of domain adaptation scenario where the Teacher network is trained using UK-DALE, while the Student is trained using REFIT (UK-DALE-to-REFIT scenario). The proposed explainability guided learning led to an increase in performance compared to the baseline model for all appliances. When comparing with the Teacher model, there are improvements for all appliances, except for WD, where the F-score remains unchanged, and KT, where the F-score decreased, but still remained significantly higher than the baseline model. A possible reason for the poor performance for KT is the fact that in this case, the Teacher model might not be ideal for knowledge distillation, as its low recall value suggests that it exhibits a high number of false negative predictions. Results for the domain adaptation scenario where both Teacher and Student models were trained using REFIT data (REFIT-to-REFIT) and tested on unseen houses in REFIT are shown in Table 12.4. As in the first scenario, improvements are reported in the performance compared to the baseline and the Teacher, with the exception of MW, where all three methods provide similar performance. Results presented in Figure 12.1 suggest that explainability guided learning helps alleviate incorrect transfer of explanation knowledge, and through this process improves the predictive performance of the Student model. The results presented in Tables 12.3 and 12.4 show that the proposed explainability guided learning leads to improved knowledge distillation for most appliances in both domain adaptation scenarios. In the first scenario, there are improvements of the F-Score measure ranging from 1.6% (for DW) up to 22.6% (for WM) compared to the baseline, while the improvements over the Teacher model ranged from 0% (for WD) up to 33.3% (for MW). Similar findings hold for the REFIT-to-REFIT domain adaptation scenario, where the maximum improvement over the baseline was 15.6% (for WD), while the maximum improvement over the Teacher was 25.5% (for DW). Moving from UK-DALE-to-REFIT domain, transferability has a strong influence on presence of false positive activations, while this phenomenon is not impacting the REFIT-to-REFIT scenario as much, and precision is increased more consistently as a result.

Appliance	Model	Precision	Recall	F1-Score
Washing Machine	Teacher	**0.56**	0.69	0.62
	Baseline	0.70	0.43	0.53
	Ours	0.55	**0.81**	**0.65**
Dishwasher	Teacher	0.49	0.84	0.62
	Baseline	0.50	**0.88**	0.63
	Ours	**0.52**	0.83	**0.64**
Washer-Dryer	Teacher	0.79	0.77	**0.78**
	Baseline	**0.97**	0.52	0.68
	Ours	0.75	**0.81**	**0.78**
Kettle	Teacher	**0.77**	0.42	**0.55**
	Baseline	0.26	**0.98**	0.41
	Ours	0.31	0.97	0.47
Microwave	Teacher	0.43	**0.98**	0.60
	Baseline	**0.94**	0.52	0.67
	Ours	0.69	0.96	**0.80**

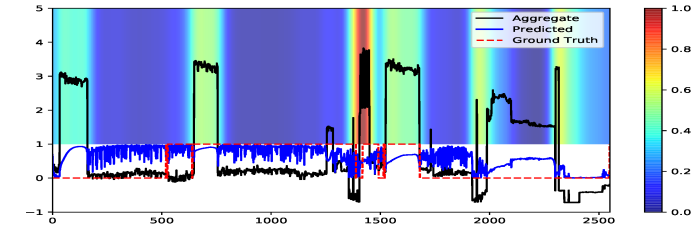
Table 12.3.: Results for the UK-DALE-to-REFIT domain adaptation scenario.

Appliance	Model	Precision	Recall	F1-Score
Washing Machine	Teacher	0.57	0.91	0.70
	Baseline	0.60	**0.93**	0.73
	Ours	**0.76**	0.82	**0.79**
Dishwasher	Teacher	0.35	**0.97**	0.51
	Baseline	0.42	0.96	0.59
	Ours	**0.49**	0.93	**0.64**
Washer-Dryer	Teacher	0.93	0.52	0.67
	Baseline	**0.98**	0.47	0.64
	Ours	0.67	**0.82**	**0.74**
Kettle	Teacher	**0.92**	0.55	0.69
	Baseline	0.60	**0.95**	0.73
	Ours	0.72	0.79	**0.75**
Microwave	Teacher	0.79	**0.98**	**0.87**
	Baseline	0.77	0.95	0.85
	Ours	**0.93**	0.77	0.84

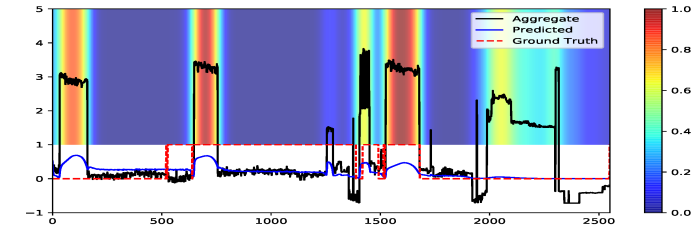
Table 12.4.: Results for the REFIT-to-REFIT domain adaptation scenario.



(a)



(b)



(c)

Figure 12.1.: Explanations for prediction of Washing Machine on a sample from the test set in the REFIT-to-REFIT domain adaptation scenario. a) Teacher explanation b) baseline Student explanation, displaying the inconsistent transfer of explanation knowledge c) Corrected Student explanation and prediction after explainability guided learning. Strong predictions are displayed before quantization.

Chapter 13.

Appliance incremental learning for Non-Intrusive Load Monitoring

A crucial aspect highlighted in [141] for NILM is the need to develop methods that can adapt to changes in the target environment and accommodate new appliances or new features of existing appliances. As largely discussed in Chapter 7, when there are discrepancies between the source and target data domains, a NILM approach that does not adjust to the ongoing changes in the deployment environment may suffer from significant performance degradation and fail to meet user expectations.

Strategies such as transfer learning [3, 138] and active learning [88] have been proposed to adapt a pre-trained model to a different data domain and minimize performance degradation due to differences between the training and target domains.

Recently, [142] proposed a continual learning approach to handle domain shift. This approach adapts the model using a limited number of data samples collected based on a specific selection criterion, and mitigates catastrophic forgetting by replaying old data during training. However, all these works focus on adapting a model to new domain characteristics post-deployment, without altering or adding new tasks, such as monitoring a new appliance.

Adjustments in the number of appliances have only been partially considered in [143, 144]. Specifically, [143] proposed a method based on a similarity criterion to identify a new appliance within the aggregate signal, but it did not focus on adaptation. Similarly, the method in [144] solely concentrates on detecting the presence of a new device.

The various neural network architectures suggested in the literature for multi-label appliance classification are all static [48, 2, 49]. This means that the quantity and type of appliances must be predetermined, and the corresponding annotated data must be available for model training. However, due to changes in user habits, this is a practical and unavoidable scenario.

In this chapter, the first work in multi-label appliance classification that suggests a method to incorporate new appliances into an already deployed network

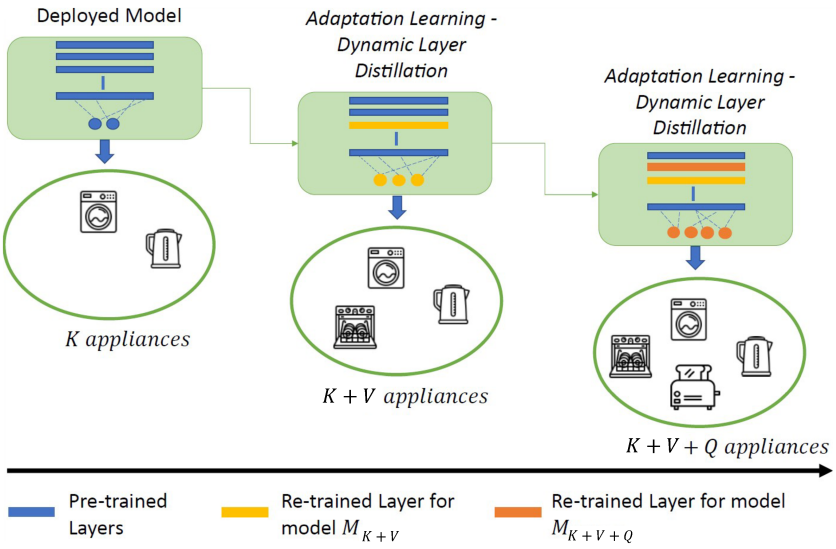


Figure 13.1.: AIL method scheme for NILM. To introduce a new appliance, adaptation learning and dynamic layer distillation are applied. The arrow from the previous to the subsequent model indicates that adaptation learning is done by using the previous model as the Teacher in the distillation. In this case, V and Q are equal to 1.

will be described. Furthermore, the method proposed a learning approach to alleviate catastrophic forgetting while enhancing the performance of new appliances. This approach employs distillation with a dynamic layer selection strategy to accomplish this goal. This method has been presented and published among the proceedings of the 14th IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids 2023 [145].

13.1. Proposed Methodology

Multi-label appliance classification aims to detect the state of K appliances at the same time, given only the power reading of the mains $y(t)$. The proposed method considers that $\tilde{K}$ can increase, and J appliances from K can be included in a subsequent monitoring phase so that $\tilde{K}+J$ is the new number of appliances to be monitored.

13.1.1. Neural Network Architecture

Multi-label appliance classification is addressed by using a CRNN, described in Section 5.1.1. The network model denoted as $M_{\tilde{K}}^{\mathbf{\Phi}}$ has trainable parameters $\mathbf{\Phi}$ and the last fully connected layer composed of $\tilde{K}$ neurons. Thus, by adding J new appliances, the new model will have $\tilde{K}+J$ neurons and will be denoted as $M_{\tilde{K}+J}^{\mathbf{\Theta}}$ with trainable parameters $\mathbf{\Theta}$.

The learning part is based on the Knowledge Distillation strategy [70] where the model $M_{\tilde{K}}^{\mathbf{\Phi}}$ is considered as the Teacher while the new model $M_{K+V}^{\mathbf{\Theta}}$ plays the role of the Student. proposed approach takes as input a series of disjointed aggregate windows of dimension L producing a series of disjointed windows of predictions, for each appliance. Figure 13.1 shows the overall framework and in the following sections, each part will be described.

13.2. Appliance-Incremental Learning Framework

In real-world scenario, new appliances can be introduced or removed. Also, already existing appliances can be used more frequently and then there could rise the necessity to start monitoring this new usage habit. The situation where appliances are removed is ignored, and the focus is exclusively on integrating the new appliances classification in the already deployed NILM algorithm, limiting an eventual performance degradation for the previous learned tasks.

13.2.1. Baseline learning

The model $M_{\tilde{K}}^{\mathbf{\Phi}}$ is trained by using the Binary Cross-Entropy (BCE) loss estimated on the predictions and the labels about the appliances' states of activations. The BCE loss is defined as Equation 5.7 with L equal to the number of samples in the input windows, and $\hat{s}_k(t)$ the state prediction.

13.2.2. Adaptation learning

The learning strategy adopted to avoid *catastrophic forgetting* while adapting the network to monitor new appliances is based on the Teacher-Student knowledge distillation principle [70]. Exploiting the knowledge of the Teacher model $M_{\tilde{K}}^{\mathbf{\Phi}}$, the new model $M_{\tilde{K}+J}^{\mathbf{\Theta}}$ is forced to preserve the performance related to the initial $\tilde{K}$ appliances while learning to classify additional V appliances. In this way, exploiting the Teacher knowledge, signals and annotations storage related to the initial $\tilde{K}$ appliances are avoided. The Student model, $M_{\tilde{K}+V}^{\mathbf{\Theta}}$, is initialized with the weights of the Teacher model $M_{\tilde{K}}^{\mathbf{\Phi}}$ (i.e., initially $\mathbf{\Theta}=\mathbf{\Phi}$), and V neurons are added to the final fully connected layer, which are then initialized

randomly. The distillation loss L_d is defined as the BCE computed on the $M_K^{\mathbf{\Phi}}$ and the $M_{\tilde{K}+J}^{\mathbf{\Theta}}$ models predictions respectively with $\hat{s}_{k,T}(t)$ and $\hat{s}_{k,S}(t)$:

$$L_d^{\mathbf{\Theta}} = -\frac{1}{\tilde{K}}\frac{1}{L}\sum_{k=1}^{\tilde{K}}\sum_{t=1}^{L}[\hat{s}_{k,T}(t)\log(\hat{s}_{k,S}(t)) + (1-\hat{s}_{k,T}(t))\log(1-\hat{s}_{k,S}(t))], \tag{13.1}$$

Note that the loss is calculated only for the first $\tilde{K}$ appliances.

On the other hand, the Student model $M_{\tilde{K}+V}^{\mathbf{\Theta}}$ needs to be trained to classify the new V appliances. Thus, the second component of the loss function is the BCE computed on the labels collected in the target building just for the new appliances and the network predictions as follow:

$$L_{new}^{\mathbf{\Theta}} = -\frac{1}{V}\frac{1}{L}\sum_{j=\tilde{K}+1}^{V+\tilde{K}}\sum_{t=1}^{L}[s_v(t)\log(\hat{s}_{v,S}(t)) + (1-s_v(t))\log(1-\hat{s}_{v,S}(t))]. \tag{13.2}$$

The final loss function used to train the new $M_{K+V}^{\mathbf{\Theta}}$ model is formulated as:

$$L_{AIL}^{\mathbf{\Theta}} = \alpha L_d^{\mathbf{\Theta}} + (1-\alpha)L_{new}^{\mathbf{\Theta}} \tag{13.3}$$

where α is the weight to balance the loss contributions. After calculating $L_{AIL}^{\mathbf{\Theta}}$, proposed method incorporates a layer selection phase to optimize the best layer parameters and improve adaptation effectiveness. The following section describes the dynamic layer selection procedure.

13.2.3. Dynamic Layer Selection

Following the idea proposed in [146, 147], there is no investigation on which is the best layer to be updated based on its performance at the end of the learning process. Instead, the proposed approach involves selecting the optimal layer for updating during each training batch. This method is incorporated into the distillation process to evaluate layer significance for both the previous K and new appliances J. For this reason, it is defined as Dynamic Layer Distillation (DLD). Differently from previous works [146, 147], the approach takes into account the loss value $L_{AIL}^{\mathbf{\Theta}}$, which encompasses the contributions from both the old and new tasks. In this way, only the layers that are at the same time less significant for the previous task and more important for the new appliances are trained. As well, there is no retraining of the layers that hold significant knowledge from the previous task, minimizing the risk of forgetting previous knowledge.

Algorithm 4 Pseudo-code outlining the proposed Dynamic Layer Distillation (DLD) method.

Require: Teacher $M_{\tilde{K}}$ model designed for $\tilde{K}$ appliances, V additional appliances
  $\boldsymbol{\Theta} \leftarrow \boldsymbol{\Phi}$;
  for e in *epochs* **do**
    for each minibatch B **do**
      Calculate $L_{AIL}^{\boldsymbol{\Theta}}$ based on Equation Equation 13.3;
      for $p \in \{C1, C2, C3, GRU\}$ **do**
        $\boldsymbol{\Theta}_p \leftarrow \boldsymbol{\Theta}$;
      end for
      Calculate the gradient $\nabla L_{AIL}^{\boldsymbol{\Theta}}$;
      Update $\boldsymbol{\Theta}_{C1}$, $\boldsymbol{\Theta}_{C2}$, $\boldsymbol{\Theta}_{C3}$, $\boldsymbol{\Theta}_{GRU}$ using Adam optimizer;
      $\boldsymbol{\Theta} \leftarrow \underset{\boldsymbol{\Psi} \in \{\boldsymbol{\Theta}_{C1}, \boldsymbol{\Theta}_{C2}, \boldsymbol{\Theta}_{C3}, \boldsymbol{\Theta}_{GRU}\}}{\arg\min} L_{AIL}^{\boldsymbol{\Psi}}$;
    end for
  end for

In accordance with Section 13.1.1, $\boldsymbol{\Theta}$ encompasses all the trainable parameters of the Student model. Specifically, $M_{\tilde{K}+V}^{\boldsymbol{\Theta}_p}$ is the particular Student model where only the parameters related to layer p and the last fully connected layer are trainable, while the remaining are held fixed. According to the architecture of the CRNN employed in this study $p \in \{C1, C2, C3, GRU\}$ where $C1$, $C2$, and $C3$ correspond to the first, second, and third convolutional layer, respectively, and GRU represents the bidirectional layer.

As shown in Algorithm 4, for each minibatch, the gradient of $L_{AIL}^{\boldsymbol{\Theta}}$ is computed, and then applied it individually to each trainable layer of Student models $M_{\tilde{K}+V}^{\boldsymbol{\Theta}_p}$, with $p \in \{C1, C2, C3, GRU\}$. Subsequently, the Student model and the related parameters that yield the minimum $L_{AIL}^{\boldsymbol{\Theta}}$ after gradient update are determined. The selected layer and the related weights are then used to initialize the current best model $M_{\tilde{K}+V}^{\boldsymbol{\Theta}}$ as starting point for the subsequent training step. This procedure is iterated over a certain number of epochs until the stopping criterion is satisfied.

Since the lowest value of $L_{AIL}^{\boldsymbol{\Theta}}$ is determined on each batch, the batch size influences the selection process and the small the size, the specific will be the selection for the windows contained in the batch. In the case where the batch size is set to 1, the layer selection is determined by the value of $L_{AIL}^{\boldsymbol{\Theta}}$ for each input window. In the experimental section, the influences that different batch sizes have on the performance are studied.

13.3. Experiments

13.3.1. Dataset

The experiments have been carried out by using the UK-DALE [24] and REFIT datasets [25]. For the datasets details refer to Appendix Section 1. The appliances considered in the experiments depend on the devices included in the test target houses which are Kettle (KE), Washing Machine (WM), Dishwasher (DW), and Toaster (TOA) for House 2 and Kettle and Microwave (MW) for House 4.

During the adaptation phase, only one month of annotated data is used from each target house for each appliance. The composition of the adaptation dataset is presented in Table 13.1.

13.3.2. Evaluation metrics

Two metrics have been used to evaluate the proposed approach. The first is the F_1-score (F_1) and the related micro-average already defined in Equation 5.9 and Equation 5.10.

13.3.3. Hyperparameters

Several hyperparameters that play a crucial role in both the learning and layer selection have been investigated.

Initially, α was set to 0.4 in Equation 13.3. Then, during the second distillation phase, α is set to 0.5. Different batch sizes are explored, since the smaller the size the more specific the layer selection for each instance. An example of the performance trend with batch sizes varying from 1 to 16 is reported in Figure 13.2 for the model M_{2+1}. Performance significantly degrades for all the appliances when the batch size is set to 16. However, for smaller values, the F1-scores remain relatively similar. The best performance on average is obtained when the batch size is equal to 1.

The input window is of 2550 samples and the maximum number of epochs to 1000 and used early stopping with a patience of 5 to avoid over-fitting.

Table 13.1.: Re-training data characteristics in terms of active samples.

Appliances	**House 2**	**House 4**
Dishwasher	189375	Not present
Toaster	4646	Not present
Microwave	Not present	9866

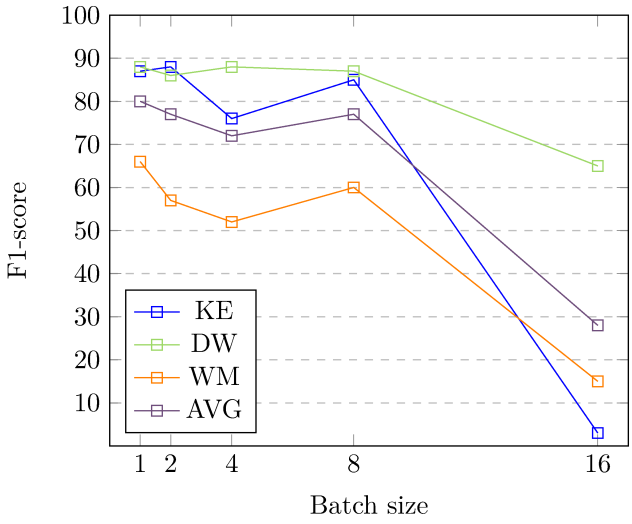


Figure 13.2.: Performance trend with varying batch sizes for the proposed AIL approach. AVG denotes the average performance.

13.3.4. Experimental procedure

The proposed method is evaluated in a real-world scenario where public aggregate and appliance-level data are available. This scenario is represented by UK-DALE and it is used to create the so-called Deployed model $M_{\tilde{K}}$, because it is initially deployed in the target houses. $M_{\tilde{K}}$ is designed to classify $\tilde{K} = 2$ appliances, specifically KE and WM, which are present in both REFIT target houses. Then, M_2 is adapted to each target house adding the new appliances of interest for that particular house. This adaptation is achieved by using one month of data belonging to target houses of REFIT. In a real-world scenario, this data would have been collected locally, i.e., from the users' houses. Thus, it is worth limiting the quantity of data to be collected and annotated. The choice of these target houses has been done to explore: (i) when the appliances monitored by the deployed model are both present in the target house (House 2) (ii) when only one appliance of the deployed model is present (House 4). Moreover, it is possible to evaluate the effect of the adaptation for three different appliances. Firstly the performance of the approach are evaluated when introducing one appliance in the monitoring (from M_2 to M_{2+1} for Houses 2 and 4). Then, the evaluation of the performance is computed when adding a second appliance considering two cases: (i) the M_{2+1} became the Teacher for the Student M_{2+1+1}, representing the case where the appliances are added progressively (ii) two appliances are added thus the Teacher is M_2. To thoroughly evaluate the method, also the models M_3 and M_4 where all appliances to be

monitored are known before the deployment were trained. This allows us to compare the performance of the method against scenarios where all appliances are pre-determined. M_3 and M_4 models are trained with UK-DALE for KE and WM and data from the target environment are used only for the new appliances. The proposed method is referred to as Proposed-AIL, while the Learning without Forgetting approach is referred to as LwF. The models M_3 and M_4 are denoted as Static. Additionally, the results related to the initial model Deployed are reported, assumed to be deployed in the target environment. The Proposed-AIL code is available on GitHub[1].

13.4. Results

The obtained results are shown in Table 13.2 and Table 13.3. Firstly, the M_2 (Deployed) model performance is reported to assess the performance of Student models after adaptation. Considering the results for House 2 in Table 13.2, when the DW is introduced, the Precision and Recall of the M_3 Static approach for the KE and WM decrease compared to M_2 Deployed. This trend is also observed for the M_{2+1} LwF approach and for M_{2+1} Proposed-AIL only for WM. This evidences that for the WM, other factors could influence the classification, like the introduction of an appliance with similar characteristics as highlighted by the scores obtained for all the evaluated methods. For KE, instead, with proposedapproach there is a slight improvement compared to the initial M_2 Deployed model. In summary, when introducing the DW, compared to LwF, the Proposed-AIL better mitigates the forgetting both for KE and WM. Regarding the DW, the new appliance, both LwF and Proposed-AIL achieve similar performance with respect to the M_3 Static approach, that presents the highest F_1-score. On average, the proposed approach obtained higher F_1-score than M_3 Static and M_{2+1} LwF methods.

When the model is adapted to classify an additional appliance, the TOA, the M_4 Static model exhibits an improved Precision for KE, decreased Recall for WM, and stable performance for DW compared to the M_3 Static model. The M_{2+1+1} LwF shows an improvement for KE compared to M_{2+1} LwF. Nonetheless, comparing the models that introduced the TOA, KE, WM, and TOA are better classified by M_{2+1+1} Proposed-AIL while the DW is better classified by M_{2+1+1} LwF. On average, proposed method outperforms both M_4 Static and M_{2+1+1} LwF.

In the case where both DW and TOA are introduced together, proposed method consistently outperforms both the M_4 Static and M_{2+2} LwF approaches, showing the best overall results. For all the appliances, proposed method presents an improvement in terms of Precision, Recall, and F1-score compared

[1] `https://github.com/GiuTan/ApplianceIncrementalLearning-NILM`

Table 13.2.: Results related to House 2. Best F_1-score when considering the same appliances are highlighted in bold.

			Appliance				
Model	**Approach**	**Metric**	**KE**	**WM**	**DW**	**TOA**	**AVG**
M_2	Deployed	PR	0.86	0.65	-	-	0.76
		RE	0.87	0.89	-	-	0.88
		F1	0.87	0.75	-	-	0.81
M_3	Static	PR	0.65	0.50	0.85	-	0.67
		RE	0.89	0.81	0.90	-	0.87
		F1	0.75	0.62	**0.87**	-	0.75
M_{2+1}	LwF[78]	PR	0.78	0.55	0.94	-	0.77
		RE	0.78	0.74	0.80	-	0.77
		F1	0.78	0.63	0.86	-	0.76
	Proposed-AIL	PR	0.89	0.54	0.81	-	0.75
		RE	0.87	0.86	0.90	-	0.88
		F1	**0.88**	**0.66**	0.86	-	**0.80**
M_4	Static	PR	0.80	0.52	0.84	0.54	0.68
		RE	0.82	0.72	0.91	0.63	0.77
		F1	0.81	0.61	0.87	0.58	0.72
M_{2+1+1}	LwF[78]	PR	0.85	0.47	0.88	0.85	0.76
		RE	0.82	0.81	0.92	0.67	0.81
		F1	0.83	0.60	**0.90**	0.75	0.77
	Proposed-AIL	PR	0.88	0.53	0.84	0.84	0.77
		RE	0.84	0.83	0.90	0.72	0.82
		F1	**0.86**	**0.64**	0.87	**0.78**	**0.79**
M_{2+2}	LwF[78]	PR	0.81	0.53	0.89	0.78	0.75
		RE	0.84	0.86	0.93	0.69	0.83
		F1	0.83	0.66	**0.91**	0.73	0.78
	Proposed-AIL	PR	0.89	0.56	0.89	0.82	0.79
		RE	0.87	0.87	0.92	0.72	0.85
		F1	**0.88**	**0.68**	**0.91**	**0.77**	**0.81**

to the M_4 Static. Compared to M_{2+2} LwF, proposed method presents higher scores, except for DW which has the same Precision and a slightly lower value for the Recall. Due to rounding, the F_1-scores of the two methods are the same.

The results presented in Table 13.3 show that, on average, the proposed method outperforms both the LwF and M_3 Static approaches in terms of F_1-score, even when evaluated on House 4 data. When MW is introduced, the M_{2+1} Proposed-AIL model produces fewer false positives for KE than all the other models, as shown by the Precision value. Nonetheless, compared to M_2 Deployed and M_3 Static, proposed approach exhibits a decreased Recall for KE, suggesting an underestimation of the activations. For MW instead, the proposed method achieves the highest Recall.

Table 13.3.: Results related to House 4. Best F_1-score when considering the same appliances are highlighted in bold.

			Appliance		
Model	**Approach**	**Metric**	**KE**	**MW**	**AVG**
M_2	Deployed	PR	0.57	-	0.57
		RE	0.89	-	0.89
		F1	0.70	-	0.70
M_3	Static	PR	0.52	0.50	0.51
		RE	0.96	0.90	0.93
		F1	**0.68**	0.64	0.66
M_{2+1}	LwF[78]	PR	0.61	0.93	0.77
		RE	0.51	0.92	0.72
		F1	0.56	**0.93**	0.75
	Proposed-AIL	PR	0.63	0.90	0.77
		RE	0.62	0.93	0.78
		F1	0.62	0.92	**0.77**

Based on the presented results, it is evident that adapting the pre-trained network as in LwF and Proposed-AIL is effective in mitigating catastrophic forgetting and achieving better overall performance, compared to including target data in the pre-training set and re-training the model (as in M_3 Static and M_4 Static). The performance variations observed after introducing a new appliance can also be influenced by the nature of the appliance itself, as demonstrated in the cases of WM and DW, or KE and TOA. Due to the complex interactions among appliances, further investigations are required to fully understand these dynamics.

Nonetheless, applying the proposed method is advantageous when introducing one or two appliances simultaneously. The dynamic layer selection technique is particularly effective in finding a suitable balance between the knowledge of previous and new appliances, surpassing the limitations of training all layers as in the LwF approach.

Chapter 14.

Discussion

This part of the thesis offered a detailed description of three approaches that look at the direction of low-complexity deep learning algorithms for NILM.

The approach based on weakly supervised knowledge distillation has been proposed to enhance the scalability of the NILM approach on edge devices with low resources. Starting from a large pre-trained model, the network is adapted to the target scenario data while reducing the complexity. Gradually evaluating network layers removal allowed to a better understanding of performance degradation or improvement when occurred. Evaluated in two different practical scenarios, the method reduced the number of network parameters up to 10 times compared to the initial model while maintaining performance. An extension of method has been presented in order to eliminate inconsistencies between the Teacher and the Student networks behaviour during the training. The inconsistencies have been detected by using an explainability tool that shown where the network attention is focusing when producing the inference. The explainability maps are included as an additional loss term, comparing the student and teacher network maps. The approaches proposed in Chapter 11 and Chapter 12 are not directly comparable because the latter approached the classification of each appliance singularly.

In the last work of this part, the appliance-incremental learning approach has been designed to minimize the number of parameters when including additional appliance monitoring into the algorithm. In this way, the NILM service provide to the user can be updated without significant additional computational resources and necessity to improve the quality of the hardware and change the device.

Future works aim to perform adaptation to new tasks by applying weak supervision and reducing the architecture of the network until reaching the best trade-off between complexity and performance. Moreover, a lightened training strategy should be adopted to allow possible adaptation on local devices preserving more user privacy.

Part IV.

Conclusions

Chapter 15.

Conclusions and future works

NILM is an effective tool to monitor appliance-level consumption and thus to promote energy awareness, especially in residential settings. From consumption awareness, applications and services can be extended to support users and utilities in EMS and DR programs. A huge quantity of research effort has been placed until now on this topic and deep learning approaches have been largely applied, representing the state-of-the-art. This thesis addressed three major gaps in the current NILM literature.

Firstly, supervised learning methods are often hindered by the need for labeled datasets, which are challenging to acquire in real-world scenarios. To obtain this, in the first part, this thesis proposed several strategies to simplify the role of the end-user. These strategies not only preserve the role of the user, but also improve performance compared to state-of-the-art methods. By exploiting coarser annotations provided by the users and developing learning strategies based on weak supervision and transfer learning, this work demonstrated that by reducing the quantity of annotated data, performance remained stable or improved. When advanced data selection strategy, as active learning, are considered to select data to be annotated by the user, the effort is even lighter while performance increases.

Due to privacy concerns associated with sensitive data, it is preferable to perform computations locally rather than on the cloud. Furthermore, issues related to latency and bandwidth can impact the quality of the provided service, especially when real-time feedback is required. Therefore, the second part of the thesis described NILM approaches designed to favor local computation, mitigating these concerns. Even with reduced complexity, these approaches enhance performance, while reducing the computational requirements more than benchmark approaches. Not only, while enriching the network functionality with additional tasks, network structure is maintained and only few additional parameters are introduced to embed the new appliances monitoring.

15.1. Future perspectives

Non-intrusive techniques are very promising, nonetheless the performance should be additionally improved to ensure the reliability and trustworthiness of the service, while preserving the role of the user and scalability, as in this thesis. A possibility to enhance performance can be the development of hybrid monitoring systems, where the majority of the appliances are monitored non-intrusively, while only few challenging appliances are monitored for a limited time with electrical sensors. This strategy could also facilitate disaggregation by exploiting the power consumption signals recorded by installed sensors to lighten the aggregate power signal [148]. In this way, the identification or disaggregation of appliances without plugs can be more accurate. Anyway, it is worth considering that the plugs still present the limitation of missing data or errors due to disconnection problems. Moreover, the installation depends on the possibility to physically plug the smart sensors on the appliance plug, if accessible. To enhance the efficiency of recently proposed methods, it might be beneficial to assess the performance of hybrid monitoring frameworks.

Looking ahead, future directions could also focus on conducting training locally allowing the network to adapt to new domains and tasks, further enhancing its applicability and effectiveness. Regarding other applications of strong and weak labels for smart energy systems, power quality disturbances identification and faults detection are tasks eligible to be modeled with weak supervision.

Motivated by the growing literature, research effort can also be placed to monitor loads in the industrial and commercial sectors that account for a large part of the total energy consumption. Particularly for the industrial scenario, energy monitoring can be effective at different levels to monitor the production, detecting energy waste and anomalies.

Finally, the outcome of the monitoring system could be exploited to develop accurate energy management systems to optimize the usage of renewable energy. Moreover, it can be integrated in advanced forecasting algorithm.

Part V.

Appendix

1. Data Preparation

The procedure to prepare the datasets is described here. Each dataset has been processed to create one sets of bags to be used for training and testing the proposed methods. The NILMTK [98] is a toolkit recently proposed to process NILM datasets and prepare them to train and test some benchmark approaches implemented in the framework. The toolkit is useful to resample and align power signals since generally the aggregate signal and the appliance consumption are acquired at different frequencies. Moreover, it is necessary to fill gaps and missing values. The procedure for creating these sets is described in the following:

1. The first step consisted in extracting the activations of the monitored appliances from the datasets. This has been performed by using NILMTK using the parameters in [28] for UK-DALE, and the ones in [25] for REFIT.

2. The second step consisted in combining the extracted activations randomly to create bags with one to four concurrent appliances. In each dataset, the maximum length of an activation is about 1500 samples, so the bag length L is set to 2550. In this way, activations can be properly placed within the segment. The location of the activation inside the bag is determined randomly. Generally, the bag length can have a role in performance; however, in the following experiments, it is important that the same value is used in all the methods considered to evaluate only the influence of weak labels.

3. The third step consisted in the extraction of the noise contribution, i.e., the term $v(t)$ in Equation 2.3. This has been obtained by selecting a random aggregate power segment of length L and then subtracting the monitored appliances' activations from it. The extracted noise term has been then summed to bags created in step two. This procedure is repeated for each bag, so noise terms are all different. Moreover, noise terms and activations of the monitored appliances always belong to the same building.

For each appliance, strong labels in a bag are set to 1 if a sample belongs to an activation (i.e., the appliance is in the ON state), and 0 otherwise. Weak labels are set to 1 if the activation of an appliance is present in the bag.

The following periods of UK-DALE were considered:

- house 1: 06/01/2016-31/08/2016;
- house 2: 01/06/2013-31/08/2013;

- house 3, 4: 16/03/2013-05/04/2013;
- house 5: 29/06/2014-05/09/2014.

As in [96], for REFIT the following date intervals are considered:

- houses 9, 12, 18: 07/12/2013-08/07/2015;
- houses 10, 17: 20/11/2013-30/06/2015;
- houses 2, 5, 7, 16: 17/09/2013-08/07/2015;
- house 13: 26/09/2013-08/07/2015;
- houses 3, 4, 6, 8, 11, 15, 19: 26/09/2013-08/07/2015.

2. Benchmark Approaches

This part of the appendix is dedicated to describe more in details the benchmark approaches used to evaluate the innovative contribution and performance improvements of the proposed methods.

2.1. Long-Short Term Memory Network

Traditional RNNs suffer of the vanishing gradient problem. Thus, the Long-Short Term Memory Network has been proposed to avoid this issue and is suitable for sequence predictions, capturing long-term dependencies. Kelly et al. [28] proposed this architecture to disaggregate appliances consumption patterns. The architecture is composed of one convolutional layer with 16 filters, a kernel size of 4, stride equal to 1 and a linear activation function. Then, the features extracted by the convolutional layer are the input of two bidirectional LSTM layers with 128 and 256 recurrent units respectively. In the end, two fully connected layer are used with 128 neurons and activation function *tanh* and 1 neuron with linear activation function.

2.2. Temporal Convolutional Network

The Temporal Convolutional Network (TCN) is a sequence modeling architecture derived from the CNNs [149]. Different from the popular canonical recurrent networks such as LSTM, TCNs do not use gating mechanisms and have much longer effective memory. The convolutional layer employees a causal convolution, meaning there is no information leakage from the future to the past, and a sequence-to-sequence approach, meaning that the input and output sequences have the same lengths as in recurrent networks. Causal convolution implies that the output at a certain step t is only convolved with elements from

that step and earlier. Dilated convolution efficiently addresses the multi-scale information integration problem. The residual connections are used to stack the dilated causal convolution layers together to form a residual block.

Regarding NILM, the approach proposed in [5] employed a TCN architecture trained by semi-supervised learning. The TCN residual block is composed by two dilated causal convolution layers with ReLU activation, weight normalization for the convolutional filters and dropout. The learning approach adopted is semi-supervised and applied with the Teacher-Student architecture: in this context, one network learns from labeled data while the weights of the other network are updated by using the moving average and a loss contribution is estimated comparing the output of the two networks on unlabelled data. The structure of the network is composed of:

- Number of blocks: 5
- Number of filters for each block: 64
- Filter size: 3
- Dilation factor: 2^i for block i
- Activation function: ReLu
- Dropout probability: 0.1
- Mini batch size: 1024
- Number of epochs: 150.

The learning rate is equal to 0.002 for the Adam optimizer [94].

2.3. Sequence-to-point

The sequence-to-point approach has been largely considered in literature. It has been proposed as an alternative of the approach proposed by Kelly et al. [28] that modeled NILM as a sequence-to-sequence problem.

The main idea behind the sequence-to-point is the prediction of the middle point of the corresponding window for the target appliance, exploiting past and future samples. By using a sliding window approach, each sample of the target window is predicted. The approach is designed by using a CNN and it is trained to predict the power consumption of the appliance.

The proposed CNN is composed of:

- 5 convolutional layers
- 30, 30, 40, 50, and 50 filters for each layer respectively

- 10, 8, 6, 5, and 5 filter size for each layer respectively
- stride equal to 1
- ReLu and Linear (last layer) activations.

Bibliography

[1] International Energy Agency, "Key world energy statistics," Tech. Rep., 2021. [Online]. Available: https://www.iea.org/reports/key-world-energy-statistics-2021

[2] G. Tanoni, E. Principi, and S. Squartini, "Multilabel appliance classification with weakly labeled data for non-intrusive load monitoring," *IEEE Transactions on Smart Grid*, vol. 14, no. 1, pp. 440–452, 2023.

[3] G. Tanoni *et al.*, "Weakly supervised transfer learning for multi-label appliance classification," in *Appl. Intell. Inform.* Springer Nature Switzerland, 2022, pp. 360–375.

[4] T. Giulia, S. Tamara, P. Emanuele, S. Vladimir, S. Lina, and S. Stefano, "A weakly supervised active learning framework for non-intrusive load monitoring," *Integrated Computer-Aided Engineering*, 2024, (accepted for publication).

[5] Y. Yang, J. Zhong, W. Li, T. A. Gulliver, and S. Li, "Semisupervised multilabel deep learning based nonintrusive load monitoring in smart grids," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 11, pp. 6892–6902, 2019.

[6] Eurostat, "Eurostat statistics explained: Electricity and heat statistics," Tech. Rep., 2022.

[7] International Energy Agency, "Renewables," Tech. Rep., 2023. [Online]. Available: https://www.iea.org/reports/renewables-2023

[8] ——, "Electricity information," Tech. Rep., 2021. [Online]. Available: https://www.iea.org/data-and-statistics/data-product/electricity-information

[9] I. Vassileva and J. Campillo, "Increasing energy efficiency in low-income households through targeting awareness and behavioral change," *Renewable Energy*, vol. 67, pp. 59–63, 2014, renewable Energy for Sustainable Development and Decarbonisation.

[10] M. Benachir and C. Moulay Larbi, "Impact of household transitions on domestic energy consumption and its applicability to urban energy planning," *Frontiers of Engineering Management*, 2017.

[11] N. A. Mamoun, M. I. Zuriekat, H. I. A. Jabali, and N. A. Asfour, "Determinants of purchasing intentions of energy-efficient products: The roles of energy awareness and perceived benefits," *International Journal of Energy Sector Management*, vol. 13, pp. 128–148, 2019.

[12] L. Marchi and J. Gaspari, "Energy conservation at home: A critical review on the role of end-user behavior," *Energies*, vol. 16, no. 22, 2023.

[13] P. García Gómez, I. González-Rodríguez, and C. R. Vela, "Enhanced memetic search for reducing energy consumption in fuzzy flexible job shops," *Integrated Computer-Aided Engineering*, pp. 151–167, 2023.

[14] Y. Kabalci, "A survey on smart metering and smart grid communication," *Renewable and Sustainable Energy Reviews*, vol. 57, pp. 302–318, 2016.

[15] E. Commission, D.-G. for Energy, C. Alaton, and F. Tounquet, *Benchmarking smart metering deployment in the EU-28 : final report*. Publications Office, 2020.

[16] Z. Su *et al.*, "Secure and efficient federated learning for smart grid with edge-cloud collaboration," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1333–1344, 2022.

[17] F. Luo *et al.*, "Personalized residential energy usage recommendation system based on load monitoring and collaborative filtering," *IEEE Trans. Ind. Informat.*, vol. 17, no. 2, pp. 1253–1262, 2021.

[18] M. Zeifman and K. Roth, "Nonintrusive appliance load monitoring: Review and outlook," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 1, pp. 76–84, 2011.

[19] G. Hart, "Nonintrusive appliance load monitoring," *Proc. of the IEEE*, vol. 80, no. 12, pp. 1870–1891, 1992.

[20] F. Rossier, P. Lang, and J. Hennebert, "Near real-time appliance recognition using low frequency monitoring and active learning methods," *Energy Procedia*, vol. 122, pp. 691–696, 2017.

[21] M. Berges, E. Goldman, H. S. Matthews, L. Soibelman, and K. Anderson, "User-centered nonintrusive electricity load monitoring for residential buildings," *Journal of Computing in Civil Engineering*, vol. 25, no. 6, pp. 471–480, 2011.

[22] M. Satyanarayanan, "Pervasive computing: vision and challenges," *IEEE Personal Communications*, vol. 8, no. 4, pp. 10–17, 2001.

[23] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, p. 37–42, sep 2015.

[24] J. Kelly and W. Knottenbelt, "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes," *Scientific Data*, vol. 2, no. 150007, 2015.

[25] D. Murray, L. Stankovic, and V. Stankovic, "An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study," *Scientific Data*, vol. 4, no. 1, p. 160122, 2017.

[26] J. Z. Kolter and M. J. Johnson, "REDD: A public data set for energy disaggregation research," in *Proc. of Workshop on Data Mining Applications in Sustainability (SIGKDD)*, San Diego, CA, USA, 2011, pp. 59–62.

[27] S. Makonin, F. Popowich, L. Bartram, B. Gill, and I. V. Bajić, "Ampds: A public dataset for load disaggregation and eco-feedback research," in *2013 IEEE Electrical Power & Energy Conference*, 2013, pp. 1–6.

[28] J. Kelly and W. Knottenbelt, "Neural NILM: Deep Neural Networks Applied to Energy Disaggregation," in *Proc. 2nd ACM Int. Conf. on Embedded Syst. Energy-Efficient Built Environ.*, New York, USA, Nov. 4–5 2015, pp. 55–64.

[29] C. Zhang, M. Zhong, Z. Wang, N. Goddard, and C. Sutton, "Sequence-to-point learning with neural networks for non-intrusive load monitoring," in *Proc. of the 32nd AAAI Conf. on Artificial Intelligence*, New Orleans, LA, USA, Feb. 2-7 2018, pp. 2604–2611.

[30] M. Kaselimi, E. Protopapadakis, A. Voulodimos, N. D. Doulamis, and A. D. Doulamis, "Multi-channel recurrent convolutional neural networks for energy disaggregation," *IEEE Access*, vol. 7, pp. 81 047–81 056, 2019.

[31] M. Xia, K. Wang, X. Zhang, Y. Xu *et al.*, "Non-intrusive load disaggregation based on deep dilated residual network," *Electric Power Systems Research*, vol. 170, pp. 277–285, 2019.

[32] Y. Zhang, W. Qian, Y. Ye, Y. Li, Y. Tang, Y. Long, and M. Duan, "A novel non-intrusive load monitoring method based on resnet-seq2seq networks for energy disaggregation of distributed energy resources integrated with residential houses," *Applied Energy*, vol. 349, p. 121703, 2023.

[33] A. Langevin, M.-A. Carbonneau, M. Cheriet, and G. Gagnon, "Energy disaggregation using variational autoencoders," *Energy and Buildings*, vol. 254, p. 111623, 2022.

[34] M. Kaselimi, N. Doulamis, A. Voulodimos, A. Doulamis, and E. Protopapadakis, "EnerGAN++: A Generative Adversarial Gated Recurrent Network for Robust Energy Disaggregation," *IEEE Open Journal of Signal Processing*, vol. 2, p. 1, 2021.

[35] Y. Pan, K. Liu, Z. Shen, X. Cai, and Z. Jia, "Sequence-to-subsequence learning with conditional GAN for power disaggregation," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*, Barcellona, Spain, May 4-8 2020, pp. 3202–3206.

[36] N. Virtsionis-Gkalinikis, C. Nalmpantis, and D. Vrakas, "Saed: self-attentive energy disaggregation," *Machine Learning*, 2023.

[37] J. Ouzine, M. Marzouq, and S. Dosse Bennani, "New hybrid deep learning models for multi-target nilm disaggregation," *Energy Efficiency*, 2023.

[38] R. Sun, K. Dong, and J. Zhao, "Diffnilm: A novel framework for non-intrusive load monitoring based on the conditional diffusion model," *Sensors*, vol. 23, no. 7, 2023.

[39] N. Kianpoor, B. Hoff, and T. Østrem, "Deep adaptive ensemble filter for non-intrusive residential load monitoring," *Sensors*, vol. 23, no. 4, 2023.

[40] M. Irani Azad, R. Rajabi, and A. Estebsari, "Nonintrusive load monitoring (nilm) using a deep learning model with a transformer-based attention mechanism and temporal pooling," *Electronics*, vol. 13, no. 2, 2024.

[41] L. Massidda, M. Marrocu, and S. Manca, "Non-intrusive load disaggregation by convolutional neural network and multilabel classification," *Applied Sciences*, vol. 10, no. 4, 2020.

[42] S. Verma, S. Singh, and A. Majumdar, "Multi label restricted boltzmann machine for non-intrusive load monitoring," in *Proc. of ICASSP*, 2019, pp. 8345–8349.

[43] V. Singhal, J. Maggu, and A. Majumdar, "Simultaneous detection of multiple appliances from smart-meter measurements via multi-label consistent deep dictionary learning and deep transform learning," *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 2969–2978, 2019.

[44] S. Singh and A. Majumdar, "Non-intrusive load monitoring via multi-label sparse representation-based classification," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1799–1801, 2020.

[45] L. Massidda, M. Marrocu, and S. Manca, "Non-intrusive load disaggregation by convolutional neural network and multilabel classification," *Applied Sciences*, vol. 10, p. 1454, 2020.

[46] H. Çimen, E. J. Palacios-Garcia, N. Çetinkaya, J. C. Vasquez, and J. M. Guerrero, "A dual-input multi-label classification approach for non-intrusive load monitoring via deep learning," in *Proc. of ZINC*, 2020, pp. 259–263.

[47] X. Zhou, S. Li, C. Liu, H. Zhu, N. Dong, and T. Xiao, "Non-intrusive load monitoring using a cnn-lstm-rf model considering label correlation and class-imbalance," *IEEE Access*, vol. 9, pp. 84 306–84 315, 2021.

[48] S. Verma, S. Singh, and A. Majumdar, "Multi-label lstm autoencoder for non-intrusive appliance load monitoring," *Electr. Power Syst. Res.*, vol. 199, p. 107414, 2021.

[49] L. d. S. Nolasco, A. E. Lazzaretti, and B. M. Mulinari, "DeepDFML-NILM: A New CNN-Based Architecture for Detection, Feature Extraction and Multi-Label Classification in NILM Signals," *IEEE Sensors J.*, vol. 22, no. 1, pp. 501–509, 2022.

[50] S. Singh and A. Majumdar, "Multi-label deep blind compressed sensing for low-frequency non-intrusive load monitoring," *IEEE Trans. Smart Grid*, vol. 13, no. 1, pp. 4–7, 2022.

[51] D. Li, J. Li, X. Zeng, V. Stankovic, L. Stankovic, C. Xiao, and Q. Shi, "Transfer learning for multi-objective non-intrusive load monitoring in smart building," *Applied Energy*, vol. 329, p. 120223, 2023.

[52] D. Murray *et al.*, "Transferability of Neural Network Approaches for Low-rate Energy Disaggregation," in *Proc. of ICASSP*, Brighton, UK, May 12-17 2019, pp. 8330–8334.

[53] V. Piccialli and A. M. Sudoso, "Improving Non-Intrusive Load Disaggregation through an Attention-Based Deep Neural Network," *Energies*, vol. 14, no. 4, p. 847, 2021.

[54] Y. Liu, J. Qiu, and J. Ma, "Samnet: Toward latency-free non-intrusive load monitoring via multi-task deep learning," *IEEE Transactions on Smart Grid*, vol. 13, no. 3, pp. 2412–2424, May 2022.

[55] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, L. Màrquez, C. Callison-Burch, and J. Su, Eds. Lisbon, Portugal:

Association for Computational Linguistics, Sep. 2015, pp. 1422–1432. [Online]. Available: https://aclanthology.org/D15-1167

[56] Z. Zuo, B. Shuai, G. Wang, X. Liu, X. Wang, B. Wang, and Y. Chen, "Convolutional recurrent neural networks: Learning spatial dependencies for image representation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2015.

[57] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 448–456. [Online]. Available: https://proceedings.mlr.press/v37/ioffe15.html

[58] K. Cho, B. van Merrienboer, Çaglar Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Proc. of EMNL*, 2014, pp. 1724–1734.

[59] D. Pathak, P. Krahenbuhl, and T. Darrell, "Constrained convolutional neural networks for weakly supervised segmentation," in *Proc. of the IEEE Int. Conf. on Computer Vision*, Santiago, Chile, Dec. 11-28 2015, pp. 1796–1804.

[60] Y. Wang, J. Li, and F. Metze, "A Comparison of Five Multiple Instance Learning Pooling Functions for Sound Event Detection with Weak Labeling," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Brighton, UK, May 12-17 2019, pp. 31–35.

[61] A. Kumar and B. Raj, "Audio event detection using weakly labeled data," in *Proc. of the 24th ACM Int. Conf. on Multimedia*, Amsterdam, Netherlands, Oct. 15-19 2016.

[62] N. Pappas and A. Popescu-Belis, "Explaining the stars: Weighted multiple-instance learning for aspect-based sentiment analysis," in *Proc. of the Int. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, Oct. 25-29 2014, pp. 455–466.

[63] M. Tang, Q. Zhao, and Z. Liu, "Weakly labeled semi-supervised sound event detection with multi-scale residual attention," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–7.

[64] K. Miyazaki, T. Komatsu, T. Hayashi, S. Watanabe, T. Toda, and K. Takeda, "Weakly-supervised sound event detection with self-attention," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 66–70.

[65] Y. Zhang, X. Chen, J. Li, C. Wang, and C. Xia, "Semantic object segmentation via detection in weakly labeled video," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3641–3649.

[66] M. Hu, H. Han, S. Shan, and X. Chen, "Weakly supervised image classification through noise regularization," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 11 509–11 517.

[67] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A survey of deep active learning," *ACM computing surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021.

[68] X. Yang, Z. Song, I. King, and Z. Xu, "A survey on deep semi-supervised learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 9, pp. 8934–8954, 2023.

[69] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021.

[70] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS*, 2015.

[71] B. Peng *et al.*, "Incorporating knowledge distillation into non-intrusive load monitoring for hardware systems deployment," in *Proc. of EI2*, 2021, pp. 3054–3058.

[72] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3366–3385, 2022.

[73] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," 2016.

[74] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7120–7129.

[75] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7765–7773.

[76] A. ROBINS, "Catastrophic forgetting, rehearsal and pseudorehearsal," *Connection Science*, vol. 7, no. 2, pp. 123–146, 1995.

[77] D. Lopez-Paz and M. A. Ranzato, "Gradient episodic memory for continual learning," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[78] Z. Li and D. Hoiem, "Learning without Forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, 2018.

[79] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[80] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 3987–3995.

[81] N. Miao, S. Zhao, Q. Shi, and R. Zhang, "Non-intrusive load disaggregation using semi-supervised learning method," in *Proc. of SPAC*, 2019, pp. 17–22.

[82] A. Iwayemi and C. Zhou, "Saraa: Semi-supervised learning for automated residential appliance annotation," *IEEE Transactions on Smart Grid*, vol. 8, no. 2, pp. 779–786, 2017.

[83] B. Settles, "Active learning literature survey," 2009.

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

[85] F. Liebgott and B. Yang, "Active learning with cross-dataset validation in event-based non-intrusive load monitoring," in *2017 25th European Signal Processing Conference (EUSIPCO)*. IEEE, 2017, pp. 296–300.

[86] A. M. Fatouh, O. A. Nasr, and M. Eissa, "New semi-supervised and active learning combination technique for non-intrusive load monitoring," in *Proc. of SEGE*. IEEE, 2018, pp. 181–185.

[87] L. Guo, S. Wang, H. Chen, and Q. Shi, "A load identification method based on active deep learning and discrete wavelet transform," *IEEE Access*, vol. 8, pp. 113 932–113 942, 2020.

[88] T. Todic, V. Stankovic, and L. Stankovic, "An active learning framework for the low-frequency non-intrusive load monitoring problem," *Applied Energy*, vol. 341, p. 121078, 2023.

[89] O. Maron and T. Lozano-Pérez, "A Framework for Multiple-Instance Learning," in *Advances in Neural Information Processing Systems*, vol. 10, 1998.

[90] M. A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon, "Multiple instance learning: A survey of problem characteristics and applications," *Pattern Recognition*, vol. 77, pp. 329–353, 2018.

[91] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li, "Multi-instance multi-label learning," *Artificial Intelligence*, vol. 176, no. 1, pp. 2291–2320, 2012.

[92] J. R. Foulds and E. Frank, "A review of multi-instance learning assumptions," *Knowl. Eng. Rev.*, vol. 25, pp. 1 – 25, 2010.

[93] H. K. Iqbal, F. H. Malik, A. Muhammad, M. A. Qureshi, M. N. Abbasi, and A. R. Chishti, "A critical review of state-of-the-art non-intrusive load monitoring datasets," *Electric Power Systems Research*, vol. 192, p. 106921, 2021.

[94] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of ICLR*, 12 2014.

[95] H. Dinkel, X. Cai, Z. Yan, Y. Wang, J. Zhang, and Y. Wang, "The smallrice submission to the dcase2021 task 4 challenge: A lightweight approach for semi-supervised sound event detection with unsupervised data augmentation," in *Proc. of DCASE*, 2021.

[96] L. Wang, S. Mao, B. M. Wilamowski, and R. M. Nelms, "Pre-trained models for non-intrusive appliance load monitoring," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 1, pp. 56–68, 2021.

[97] M. D'Incecco, S. Squartini, and M. Zhong, "Transfer learning for non-intrusive load monitoring," *IEEE Trans. on Smart Grid*, vol. 11, pp. 1419–1429, 2019.

[98] N. Batra *et al.*, "Towards reproducible state-of-the-art energy disaggregation," in *Proc. of BuildSys*, 2019, pp. 193–202.

[99] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *J. Mach. Learn. Res.*, vol. 18, no. 1, p. 6765–6816, 2017.

[100] "NVIDIA DGX Station A100," nvidia.com, (accessed Apr. 15, 2022). [Online]. Available: https://www.nvidia.com/en-us/data-center/dgx-station-a100/

[101] S. Makonin and F. Popowich, "Nonintrusive load monitoring (NILM) performance evaluation: A unified approach for accuracy reporting," *Energy Efficiency*, vol. 8, no. 4, pp. 809–814, 2015.

[102] S. Ray and D. Page, "Multiple Instance Regression," in *Proc. of the 18th Int. Conf. on Machine Learning*, 2001, pp. 425–432.

[103] L. Serafini, G. Tanoni, E. Principi, S. Spinsante, and S. Squartini, "A multiple instance regression approach to electrical load disaggregation," in *2022 30th European Signal Processing Conference (EUSIPCO)*, 2022, pp. 1666–1670.

[104] J. Amores, "Multiple instance classification: Review, taxonomy and comparative study," *Artificial Intelligence*, vol. 201, pp. 81–105, 2013.

[105] N. Batra, J. Kelly, O. Parson, H. Dutta, W. Knottenbelt, A. Rogers, A. Singh, and M. Srivastava, "Nilmtk: an open source toolkit for non-intrusive load monitoring," in *Proceedings of the 5th International Conference on Future Energy Systems*, ser. e-Energy '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 265–276.

[106] M. D'Incecco, S. Squartini, and M. Zhong, "Transfer learning for non-intrusive load monitoring," *IEEE Trans. Smart Grid*, vol. 11, no. 2, pp. 1419–1429, 2020.

[107] J. Lin, J. Ma, J. Zhu, and H. Liang, "Deep domain adaptation for non-intrusive load monitoring based on a knowledge transfer learning network," *IEEE Trans. Smart Grid*, vol. 13, no. 1, pp. 280–292, 2022.

[108] Y.-F. Li and Z.-H. Zhou, "Towards making unlabeled data never hurt," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 1, pp. 175–188, 2015.

[109] M. Saneii, A. Kazemeini, S. E. Seilabi, M. Miralinaghi, and S. Labi, "A methodology for scheduling within-day roadway work zones using

deep neural networks and active learning," *Computer-Aided Civil and Infrastructure Engineering*, vol. 38, pp. 1101 – 1126, 2022.

[110] Y. Yuan, F. T. K. Au, D. Yang, and J. Zhang, "Active learning structural model updating of a multisensory system based on kriging method and bayesian inference," *Computer-Aided Civil and Infrastructure Engineering*, vol. 38, no. 3, pp. 353–371, 2023.

[111] C. Klemenjak, S. Makonin, and W. Elmenreich, "Towards comparability in non-intrusive load monitoring: On data and performance evaluation," in *Proc. of ISGT*, 2020, pp. 1–5.

[112] D. Batic, G. Tanoni, L. Stankovic, V. Stankovic, and E. Principi, "Improving knowledge distillation for non-intrusive load monitoring through explainability guided learning," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[113] R. Kukunuri *et al.*, "EdgeNILM: Towards NILM on Edge devices," *BuildSys 2020 - Proc. 7th ACM Int. Conf. on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pp. 90–99, 2020.

[114] Y. Zhang *et al.*, "FedNILM: Applying Federated Learning to NILM Applications at the Edge," *IEEE Trans. on Green Commun. and Netw.*, vol. 2400, no. c, pp. 1–12, 2022.

[115] J. Barber *et al.*, "Lightweight non-intrusive load monitoring employing pruned sequence-to-point learning," *Proc. 5th Int. Workshop on Non-Intrusive Load Monitoring*, 2020.

[116] S. Sykiotis, S. Athanasoulias, M. Kaselimi, A. Doulamis, N. Doulamis, L. Stankovic, and V. Stankovic, "Performance-aware NILM model optimization for edge deployment," *IEEE Trans. Green Commun. and Netw.*, pp. 1–1, 2023.

[117] S. Ahmed and M. Bons, "Edge computed nilm: A phone-based implementation using mobilenet compressed by tensorflow lite," in *Proc. 5th Int. NILM Workshop*, 2020, p. 44–48.

[118] C. Zhang, M. Zhong, Z. Wang, N. Goddard, and C. Sutton, "Sequence-to-point learning with neural networks for non-intrusive load monitoring." AAAI Press, 2018.

[119] M. H. Phan, Q. Nguyen, S. L. Phung, W. E. Zhang, T. D. Vo, and Q. Z. Sheng, "CompactNet: A Light-Weight Deep Learning Framework

for Smart Intrusive Load Monitoring," *IEEE Sensors Journal*, vol. 21, no. 22, pp. 25 181–25 189, 2021.

[120] W. Luan *et al.*, "Leveraging sequence-to-sequence learning for online non-intrusive load monitoring in edge device," *International Journal of Electrical Power & Energy Systems*, vol. 148, p. 108910, 2023.

[121] G. Tanoni, L. Stankovic, V. Stankovic, S. Squartini, and E. Principi, "Knowledge distillation for scalable nonintrusive load monitoring," *IEEE Transactions on Industrial Informatics*, pp. 1–12, 2023.

[122] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge Distillation: A Survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021.

[123] Z. H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, 2018.

[124] J. Yim *et al.*, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proc. of IEEE CVPR*, 2017, pp. 7130–7138.

[125] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, 2020.

[126] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (gru) neural networks," in *Proc. of MWSCAS*, 2017, pp. 1597–1600.

[127] C. Klemenjak, S. Makonin, and W. Elmenreich, "Investigating the performance gap between testing on real and denoised aggregates in non-intrusive load monitoring," *Energy Informatics*, vol. 4, no. 1, 2021.

[128] W. Li, S. Gong, and X. Zhu, "Hierarchical distillation learning for scalable person search," *Pattern Recognition*, vol. 114, p. 107862, 2021.

[129] G. Yu, "Data-free knowledge distillation for privacy-preserving efficient uav networks," in *2022 6th International Conference on Robotics and Automation Sciences (ICRAS)*, 2022, pp. 52–56.

[130] M. Kaselimi, E. Protopapadakis, A. Voulodimos, N. Doulamis, and A. Doulamis, "Towards trustworthy energy disaggregation: A review of challenges, methods, and perspectives for non-intrusive load monitoring," *Sensors*, vol. 22, no. 15, p. 5872, 2022.

[131] R. Machlev, L. Heistrene, M. Perl, K. Levy, J. Belikov, S. Mannor, and Y. Levron, "Explainable artificial intelligence (xai) techniques for energy and power systems: Review, challenges and opportunities," *Energy and AI*, p. 100169, 2022.

[132] D. Murray, L. Stankovic, and V. Stankovic, "Transparent ai: explainability of deep learning based load disaggregation," in *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, 2021, pp. 268–271.

[133] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[134] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision-making and a "right to explanation"," *AI magazine*, vol. 38, no. 3, pp. 50–57, 2017.

[135] A. Jobin, M. Ienca, and E. Vayena, "The global landscape of ai ethics guidelines," *Nature Machine Intelligence*, vol. 1, no. 9, pp. 389–399, 2019.

[136] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature communications*, vol. 10, no. 1, pp. 1–8, 2019.

[137] C. J. Anders, L. Weber, D. Neumann, W. Samek, K.-R. Müller, and S. Lapuschkin, "Finding and removing clever hans: Using explanation methods to debug and improve deep models," *Information Fusion*, vol. 77, pp. 261–295, 2022.

[138] C.-H. Hur *et al.*, "Semi-supervised domain adaptation for multi-label classification on nonintrusive load monitoring," *Sensors*, vol. 22, no. 15, 2022.

[139] B. Peng, L. Qiu, T. Yu, L. Zhong, and Y. Liu, "Incorporating knowledge distillation into non-intrusive load monitoring for hardware systems deployment," in *2021 IEEE 5th Conference on Energy Internet and Energy System Integration (EI2)*, 2021, pp. 3054–3058.

[140] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618–626.

[141] M. Kaselimi *et al.*, "Towards trustworthy energy disaggregation: A review of challenges, methods, and perspectives for non-intrusive load monitoring," *Sensors*, vol. 22, no. 15, 2022.

[142] S. Sykiotis *et al.*, "Continilm: A continual learning scheme for non-intrusive load monitoring," in *Proc. of ICASSP*, 2023, pp. 1–5.

[143] J. Zhang *et al.*, "New appliance detection for nonintrusive load monitoring," *IEEE Trans. Ind. Inform.*, vol. 15, no. 8, pp. 4819–4829, 2019.

[144] X. Guo *et al.*, "Detecting the novel appliance in non-intrusive load monitoring," *Appl. Energy*, vol. 343, p. 121193, 2023.

[145] G. Tanoni, E. Principi, L. Mandolini, and S. Squartini, "Appliance-incremental learning for non-intrusive load monitoring," in *2023 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, 2023, pp. 1–6.

[146] S. Nagae *et al.*, "Automatic layer selection for transfer learning and quantitative evaluation of layer effectiveness," *Neurocomputing*, vol. 469, pp. 151–162, 2022.

[147] R. Wanjiku *et al.*, "Dynamic pre-trained models layer selection using filter-weights cosine similarity," in *Pan-African Artificial Intelligence and Smart Systems*. Springer Nature Switzerland, 2023, pp. 95–108.

[148] B. Völker, P. M. Scholls, T. Schubert, and B. Becker, "Towards the fusion of intrusive and non-intrusive load monitoring: A hybrid approach," in *Proceedings of the Ninth International Conference on Future Energy Systems*, ser. e-Energy '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 436–438. [Online]. Available: https://doi.org/10.1145/3208903.3212052

[149] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *ArXiv*, vol. abs/1803.01271, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:4747877