Machine Learning for End-use Electrical Energy Monitoring

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart
zur Erlangung der Würde eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Abhandlung

vorgelegt von
Karim Said Barsim
aus Monofiya, Ägypten

Hauptberichter:	Prof. Dr.-Ing. Bin Yang
Mitberichter:	Prof. Mario Bergés
Tag der mündlichen Prüfung:	31.05.2021

Institut für Signalverarbeitung und Systemtheorie
der Universität Stuttgart

2021

Acknowledgements

All praise is due to God by Whose grace good deeds are completed, praise which is abundant, pure, and full of blessings. Then, I gratefully thank my parents to whom I owe my greatest debt. I also owe a great debt to my wife for her great support during writing this dissertation. I thank my PhD advisor Prof. Bin Yang for his support, guidance, and for his vision throughout this work. Finally, and without enumeration, I direct my thanks to all former colleagues at the Institute of Signal processing and System theory at the University of Stuttgart and all students who decided to collaborate on this project under my supervision.

Stuttgart, 31.05.2021 Karim Said Barsim

Kurzfassung

Die Förderung des Bewusstseins der Konsumenten für ihren Energieeinsatz und -verbrauch ist eine der wichtigsten Maßnahmen zur Erreichung der Energieeffizienz in Gebäuden. Diese ist wiederum eines der Hauptziele einer klimabewussten Energiewende. Die Disaggregation und Überwachung der Endenergie ist ein praktischer und effizienter Ansatz zur gezielten Sensibilisierung der Energieverbraucher, indem ihnen in Echtzeit ein detailliertes Feedback über deren Energieverbrauch gegeben wird. Diese Arbeit befasst sich mit der Disaggregation und Überwachung der elektrischen Last beim Endverbraucher. Hierfür werden geeignete maschinelle Lernverfahren eingesetzt. Dabei wird als Erstes ein unüberwachtes (unsupervised) Disaggregationsverfahren entwickelt und validiert. Hierbei werden einfache Einschränkungen und Annahmen zugrunde gelegt, ohne dass vorklassifizierte Trainingsdaten benötigt werden. Anschließend wird ein semi-überwachte (semi-supervised) Disaggregationsverfahren entwickelt und validiert. Der angewendete Algorithmus lernt aus vorklassifizierten Daten. Er ist aber trotzdem in der Lage, die Knappheit von vorklassifizierten Daten durch den gezielten Einsatz von nicht klassifizierten Daten zu kompensieren. Zum Schluss wird eine generische neuronale Architektur für datengetriebene Disaggregation vorgestellt, die bei Verfügbarkeit von einer großen Menge an Trainingsdaten zum Einsatz kommt. Die Ergebnisse dieser Arbeit bestätigen nicht nur die Durchführbarkeit der Disaggregation der Endenergie, sondern schlagen auch effiziente Modelle vor, die sich an die Verfügbarkeit von Trainingsdaten anpassen und in der Lage sind, auf verschiedene Kategorien von elektrischen Lasten einzugehen.

Abstract

Promoting end-users awareness of their usage and consumption of energy is one of the main measures towards achieving energy efficiency in buildings, which is one of the main targets in climate-aware energy transition programs. End-use energy disaggregation and monitoring is a practical and efficient approach towards achieving the targeted awareness of energy users by providing them with real-time fine-grained feedback about their own usage of energy. In this work, we address the case of electrical energy and the problem of end-use load monitoring and disaggregation in a variety of machine learning paradigms. This work starts from unsupervised energy disaggregation based on simple constraints and assumptions without the need for labeled training data. We then study and propose semi-supervised disaggregation approaches that learn from labeled observations, but are also capable of compensating for the scarcity of labeled data by leveraging unlabeled measurements. Finally, we propose a generic neural architecture for data-driven disaggregation upon availability of an abundance of training data. Results from this work not only assert the feasibility of end-use energy disaggregation, but also propose efficient models that adapt to the availability of labeled data, and are capable of monitoring different categories of end-use loads.

Contents

Kurzfassung **v**

Abstract **vii**

List of Figures **xiii**

List of Tables **xvii**

List of Symbols **xix**

1. Introduction and Background **1**

1.1. Background . . . 2

1.2. Terminology Peculiars . . . 5

1.2.1. Intelligent or Smart ? . . . 6

1.2.2. Non-intrusive or Unsupervised ? . . . 6

1.3. Contributions . . . 7

1.4. Publications . . . 8

1.5. Thesis Outline . . . 9

1.6. Notation . . . 10

2. Load Identification under Plug-Level Monitoring **13**

2.1. Introduction . . . 14

2.2. Related Work . . . 15

2.3. Problem Statement . . . 17

2.4. Load Signatures . . . 17

2.4.1. Identifiability in Elementary Circuits . . . 18

2.4.2. VI-Waveforms as Load Signatures . . . 24

2.4.3. Training Data . . . 26

2.4.4. Data Augmentation for Translational Invariance . . . 26

2.5. Model Architecture 30
2.5.1. The Base Model 31
2.5.2. The Ensemble Architecture 33
2.5.3. Remarks 36
2.6. Performance Assessment Measures 37
2.7. Experiments and Results 40
2.7.1. The PLAID Dataset 40
2.7.2. Model Specifications 41
2.7.3. Evaluation Framework 41
2.7.4. Leveraging External Knowledge 42
2.7.5. Training under Data Scarcity 44
2.7.6. Identifiability under Reduced Sampling Rates 45
2.7.7. Temporal Stability 46
2.7.8. Related Work and Prior Art 49
2.8. Conclusion and Future Work 49

3. Unsupervised Energy Disaggregation **51**
3.1. Introduction 52
3.2. Problem Statement 54
3.3. Evaluation datasets 54
3.3.1. The residential dataset 55
3.3.2. The commercial dataset 56
3.4. Clustering-based Event Detection 56
3.4.1. Event Model 57
3.4.2. Detection Algorithm 62
3.4.3. Empirical Validation 66
3.5. Feature Extraction 67
3.6. Event Clustering 70
3.6.1. Internal Cluster Validation Measures 71
3.6.2. Empirical Validation 74
3.7. Load Recognition 76
3.7.1. Event-level Matching 76
3.7.2. Cluster-level Matching 81
3.8. Load Profile Reconstruction 83
3.9. Experiments and Results 84
3.10. Conclusion 84

4. Semi-Supervised Energy Disaggregation **89**
4.1. Introduction 90
4.2. Literature Review 91
4.3. Problem Statement 93
4.4. Model Architecture 93
4.5. Remarks 97
4.6. Performance Assessment Measures 99
4.7. Experimental Validation 99
4.7.1. Transductive Learning Experiments 100
4.7.2. Inductive Learning Experiments 101
4.8. Conclusion 104

5. Generic Deep Disaggregation **107**
5.1. Introduction 108
5.2. Related work 109
5.3. Problem Statement 112
5.3.1. Activation Profiles: Definition and Motivation 112
5.3.2. Activation Profiles: Estimation 113
5.3.3. Single load extraction 115
5.3.4. The Assumption of Independent Loads 116
5.4. Deep Disaggregation Model 119
5.4.1. Modeling Assumptions 119
5.4.2. Model Architecture 122
5.4.3. Elementary Operations 125
5.4.4. Training Procedure 127
5.4.5. Predictive Function 128
5.4.6. Remarks and Deployment Considerations 129
5.5. Time Series Performance Assessment 131
5.6. Experiments and Results 137
5.6.1. Training and Evaluation Dataset 137
5.6.2. Prior Art Validation 138
5.6.3. Long-term Validation 140
5.7. Conclusion and Outlook 146

6. Conclusion and Outlook **147**

A. Event Detection Results **149**

B. BLUED Clustering Validation **177**

C. Unsupervised Energy Disaggregation on Commercial Data: Results **179**

Bibliography **195**

List of Figures

1.1. Daily usage pattern of two loads 4

2.1. Generic LTI load 20
2.2. Examples of elementary LTI load circuits 21
2.3. Samples of VI-trajectories of end-use loads 23
2.4. Translation invariance data augmentation 30
2.5. Load identification ensemble model architecture 34
2.6. Per-load identification measures 43
2.7. Per-household load identification measures 43
2.8. Load identification under reduced training datasets 45
2.9. Load identification under sampling frequency variations 47
2.10. Transductive setting validation of SSL disaggregation 48

3.1. Unsupervised disaggregation pipeline 53
3.2. Principle of clustering-based event detection 58
3.3. The clustering-based event detection algorithm 65
3.4. Clustering-based event detection on sub-metered data 68
3.5. Internal cluster validation measures on commercial data 75
3.6. Illustration of the load recognition rules 78
3.7. Aggregate real power of the 1st phase in a commercial setting. 85
3.8. Commercial reconstructed load profile 4 86
3.9. Commercial sub-metered load profile 4 87

4.1. Self-training semi-supervised learning paradigm 97
4.2. Data splits for transductive and inductive SSL validation 100
4.3. Transductive setting validation of SSL disaggregation 102
4.4. Inductive setting validation of semi-supervised disaggregation . . . 103

5.1. Load dependency graph 118

5.2. Deep disaggregation model architecture 124
5.3. Sample aggregate real power of UK-DALE 143
5.4. WM real power and activation profile 144
5.5. WM disaggregated real power (details in text) 145

A.1. Event detection on sub-metered data 1 150
A.2. Event detection on sub-metered data 2 151
A.3. Event detection on sub-metered data 3 152
A.4. Event detection on sub-metered data 4 153
A.5. Event detection on sub-metered data 5 154
A.6. Event detection on sub-metered data 6 155
A.7. Event detection on sub-metered data 7 156
A.8. Event detection on sub-metered data 8 157
A.9. Event detection on sub-metered data 9 158
A.10.Event detection on sub-metered data 10 159
A.11.Event detection on sub-metered data 11 160
A.12.Event detection on sub-metered data 12 161
A.13.Event detection on sub-metered data 13 162
A.14.Event detection on sub-metered data 14 163
A.15.Event detection on sub-metered data 15 164
A.16.Event detection on sub-metered data 16 165
A.17.Event detection on sub-metered data 17 166
A.18.Event detection on sub-metered data 18 167
A.19.Event detection on sub-metered data 19 168
A.20.Event detection on sub-metered data 20 169
A.21.Event detection on sub-metered data 21 170
A.22.Event detection on sub-metered data 22 171
A.23.Event detection on sub-metered data 23 172
A.24.Event detection on sub-metered data 24 173
A.25.Event detection on sub-metered data 25 174
A.26.Event detection on sub-metered data 26 175

B.1. BLUED internal clustering validation 177

C.1. Aggregate real power of the 1st phase in a commercial setting. 180
C.2. Aggregate real power of the 2nd phase in a commercial setting. 181

C.3. Aggregate real power of the 3rd phase in a commercial setting. 182
C.4. Commercial reconstructed load profile 1 183
C.5. Commercial sub-metered load profile 1 184
C.6. Commercial reconstructed load profile 2 185
C.7. Commercial sub-metered load profile 2 186
C.8. Commercial reconstructed load profile 3 187
C.9. Commercial sub-metered load profile 3 188
C.10. Commercial reconstructed load profile 4 189
C.11. Commercial sub-metered load profile 4 190
C.12. Commercial reconstructed load profile 5 191
C.13. Commercial sub-metered load profile 5 192
C.14. Commercial reconstructed load profile 6 193
C.15. Commercial sub-metered load profile 6 194

List of Tables

2.1. PLAID load categories, counts, and ratios 41
2.2. PLAID per-load load identification scores 49

3.1. BLUED event detection results 67
3.2. BLUED internal clustering validation measures 74

5.1. Activation profile estimation parameters 115
5.2. UK-DALE summary statistics of selected loads 138
5.3. UK-DALE experimental comparison with prior art 139
5.4. UK-DALE long-term validation results 141

Notation and Acronyms

Notation

x	Scalar
$\hat{x}$	Estimated value of x
$\tilde{x}$	Modified value of x
x	Physical quantity or signal
$\boldsymbol{x}$	Vector
$\mathbf{X}$	Matrix
$\mathcal{X}$	Set
$x(t)$	Discrete-time signal
$\mathbf{X}_{a:b}$	Sub-sequence of the signal $\boldsymbol{x}(t)$ on the interval $a < t \leqslant b$
$\boldsymbol{x}_{a:b}$	Vectorized sub-sequence of $\boldsymbol{x}(t)$ on the interval $a < t \leqslant b$
$d(\boldsymbol{x}, \boldsymbol{x}')$	Distance function between the two points $\boldsymbol{x}$ and $\boldsymbol{x}'$
$d(\boldsymbol{x}, \mathcal{X})$	Distance function between a point $\boldsymbol{x}$ and a set $\mathcal{X}$
D_{KL}	Kullback-Leibler divergence
$(\cdot \perp\!\!\!\perp \cdot)$	Probabilistic independence
$(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$	Probabilistic conditional independence
$\exp(\cdot)$	Exponential function
$\log(\cdot)$	Natural logarithm function
$\sin(\cdot)$	Sine function
$\cos(\cdot)$	Cosine function
$\tan(\cdot)$	Tangent function
$\mathrm{diag}(\cdot)$	Diagonal matrix denotation
$\tanh(\cdot)$	Hyperbolic tangent function
$\sigma(\cdot)$	Logistic sigmoid function
$\mathrm{sgn}(\cdot)$	Sign function
$\max(a, b)$	Maximum of either the scalars a or b

$\max(\boldsymbol{x})$	Maximum element of the input vector $\boldsymbol{x}$
$\max(\mathcal{X})$	Maximum element of the set $\mathcal{X}$
$\min(\mathcal{X})$	Minimum element of the set $\mathcal{X}$
$\arg\max(\,\cdot\,)$	Argument at the minimum value
$\arg\min(\,\cdot\,)$	Argument at the maximum value
$\mathcal{N}(\mu,\ \sigma^2)$	Normal distribution with mean μ and standard deviation σ
$\mathcal{U}(a,\ b)$	Uniform distribution on the interval $[a,\ b]$
$p(\,\cdot\,)$	Probability density or mass function
$p(\,\cdot \mid \cdot\,)$	Conditional probability
$g(\,\cdot \mid \boldsymbol{\theta}_g)$	Parameterized function g with parameters $\boldsymbol{\theta}_g$
$\mathbf{1}_{N\times M}$	N-row M-column all-ones matrix
$\mathbb{1}(\cdot)$	Indicator function
$[a,\ b]$	Interval of real numbers from a to b (inclusive)
$[\![a,\ b]\!]$	Integer sequence $a, a+1, \ldots, b$

Globally used identifiers

%-NM	Percent noisy measure
acc	Accuracy or Rand accuracy
B	Informedness
$\mathbb{C}$	Set of complex numbers
dP	Change in real power
dQ	Change in reactive power
F1S	F_1-score
$\boldsymbol{F_g}$	Grid- or Power-line- frequency
FN	Number of false negatives
fn	False negatives ratio to number of samples
FP	Number of false positives
fp	False positives ratio to number of samples
$\boldsymbol{F_s}$	Sampling frequency
i(t)	Instantaneous current signal
M	Markedness
MCC	Matthews correlation coefficient

$\mathbb{N}$	Set of natural numbers $\{1, 2, 3, \cdots\}$
$\mathbb{N}_{\leqslant a}$	The set of natural numbers up to a inclusive $\{1, \cdots, a\}$
$\mathtt{P}^{(m)}(t)$	Segregated real power signal of the m^{th} load
$\mathtt{P}(t)$	Real power signal
$\mathtt{p}(t)$	Instantaneous power signal
PN	Number of negative predictions
pn	Negative prediction marginal
PP	Number of positive predictions
pp	Positive prediction marginal
$\mathbb{Q}$	Set of rational numbers
$\mathtt{Q}(t)$	Reactive power signal
$\mathbb{R}$	Set of real numbers
$\mathbb{R}_{>0}$	Set of positive real numbers
$\mathbb{R}_{\geqslant 0}$	Set of non-negative real numbers
RN	Number of real negatives
rn	Real negatives prior
RP	Number of real positives
rp	Real positives prior
tn	True negatives ratio to number of samples
TN	Number of true negative
TNA	True negative accuracy (aka inverse-precision)
TNR	True negative rate (aka inverse-recall)
TP	Number of true positives
tp	True positives ratio to number of samples
TPA	True positive accuracy (aka precision)
TPR	True positive rate (aka recall)
$\boldsymbol{T_s}$	Sampling period
$\mathtt{v}(t)$	Instantaneous voltage signal
$\mathbb{Z}$	Set of integers
$\mathbb{Z}_{\geqslant 0}$	Set of non-negative integers

Mathematical operations

$\sum$	The sum operator
$\prod$	The product operator
$[\cdot]^{\mathsf{T}}$	Matrix transpose
$\cdot * \cdot$	Convolution
$\mathfrak{T}_{\tau}$	Shift or translation operator
$\mathfrak{T}_{\tau}^{-1}$	Inverse translation operator
$\mathfrak{R}\{x\}$	Real part of the complex valued scalar x
vec($\mathbf{X}$)	Vectorization of the matrix $\mathbf{X}$
tr($\mathbf{X}$)	Trace of a matrix $\mathbf{X}$
$\lvert x \rvert$	Absolute value of the scalar x
$\lvert \mathcal{X} \rvert$	Cardinality of the set $\mathcal{X}$
$\lvert \underline{x} \rvert$ $\lvert \boldsymbol{X} \rvert$	Element-wise absolute value of a vector $\underline{x}$ or a matrix $\boldsymbol{X}$
$\lVert \boldsymbol{x} \rVert_2$	L_2 -norm of a vector $\boldsymbol{x}$

Acronyms and Abbreviations

A	Ampere
AC	Alternating current
AC	Air conditioner
AIC	Akaike information criterion
ANN	Artificial neural network
BL	Boiler
BLUED	Building-level fully labeled energy disaggregation dataset
BN	Batch normalization
BNN	Bayesian neural network
CFL	Compact fluorescent lamp
cGAN	Conditional generative adversarial network
CNN	Convolutional neural network
CONV	Dilated temporal convolutions
dAE	De-noising auto-encoder
$\mathtt{DB_I}$	Davies-Bouldin index

DBSCAN	Density-based spatial clustering for applications with noise
D_I	Dunn's index
DT	Decision trees
DTW	dynamic time warping
DW	Dishwasher
EM	Expectation maximization
F	Farad
fHMM	factorial hidden Markov model
FIR	Finite impulse response
FR	Fridge
FSM	Finite state machine
GLR	Generalized likelihood ratio
GN	Gaussian activation noise
H	Henry
HMM	Hidden Markov models
Hz	Hertz
i.i.d	Independent and identically distributed
IoT	Internet of things
KFDA	Kernel Fischer discriminant analysis
kHz	Kilo Hertz
k-NN	*k*-nearest neighbours
KT	Kettle
kW·H	Kilo-Watt hour
LC	Lighting circuit
LogSg	Logistic sigmoid function
LReLU	Leaky rectified linear unit
LSTM	Long short term memory
LTI	Linear time-invariant
MC	Microwave oven
μF	Micro-Farad
mH	Milli-Henry
ML	Maximum likelihood
MLP	Multi-layer perceptron
MSE	Mean-squared-error
NILM	Non-intrusive load monitoring

Ω	Ohm
PDF	Probability density function
PLAID	Plug-level appliance identification dataset
PMF	Probability mass function
RBF	Radial basis function
REDD	Reference energy disaggregation dataset
RFT	Random forest tree
RMS	Root mean square
RNN	Recurrent neural neural
ResNet	Residual Network
SCP	Switch continuity principle
SMPS	Switching mode power supply
SP	Solar thermal pump
SSE	Sum of squared errors
SSL	Semi-supervised learning
SVM	Support vector machine
TS	Toaster
TV	Television
UK-DALE	UK-domestic appliance-level electricity
V	Volt
VI	Voltage and current
WHITED	Worldwide household and industry transient energy dataset
WM	Washing machine
XB_I	Xie-Beni index

Chapter 1.

Introduction and Background

Many countries worldwide have started adopting counter measures for the risk of limited fossil energy resources facing the vast and growing global energy demand [BP 2019]. Among these measures is the transition to renewable energy resources [REN21 2019], as for example what is adopted in Germany under the *Energiewende* (energy transition) program [Jacobs 2012]. In 2017, *Energiewende* has managed to redirect more than third of the gross electricity consumption to renewable energy resources, and is aiming for a more ambitious 80% transition by 2050, coupled with 80-95% reduction in greenhouse gas emissions compared to 1990 and 80% savings in energy consumption in buildings [for Economic Affairs and Energy BMWi]. Transition to the fluctuating renewable energy resources poses real challenges to ensure a secure and sustainable energy supply. Additionally, "The Energy for the Future" (the monitoring program of *Energiewende*) reported that reductions in buildings' energy consumption in 2016 were five times slower than formerly targeted for 2020 [Bayer 2015, for Economic Affairs and Energy BMWi]. Involving end-users in the energy efficiency policies and promoting their awareness of their own energy usage remains one of the key measures in reaching energy efficiency targets [EUD 2012; 2020].

The need for empowering end-users with a promoted awareness of their energy usage is inevitably rising as they continue to represent a central participant in the energy market. Residential end-users in Germany, as an example, account for over a quarter of the gross electricity consumption [der Energie-und Wasserwirtschaft BDEW], and represent the largest number of customers in the German energy market [Bayer 2015].

On a parallel track, and with the era of digitisation, large scale smart meter roll-out plans began to appear in many countries [Cooper and Shuster 2019, Jenkins and Hopkins 2018, BEIS 2018, for Economic Affairs and Energy a]. In 2016, for example, and with the Metering and Digitisation of Energy Transition Acts coming into force [for Economic Affairs and Energy a], the German power system promoted expectations of a large scale roll-out of smart and intelligent metering systems [for Economic Affairs and Energy b].

Motivated by the goals for energy efficiency in buildings, either during or post the transition to renewable energies, and the evident role of private consumers in achieving these goals, we present in this work a variety of end-use monitoring and disaggregation approaches. The main objective of end-use energy monitoring is to promote public awareness of the energy consumption in buildings by leveraging the digitisation of energy transition. In this work, we investigate capabilities, challenges, and limits of these monitoring and disaggregation approaches, and propose practical and cost-effective solutions under the framework of what is referred to as *non-intrusive load monitoring*.

1.1. Background

Energy disaggregation or non-intrusive load monitoring[1] (NILM) refers to the set of computational techniques whose objective is to infer fine-grained electric load profiles (that is, time series power draws), normally at the end-use level, from a set of aggregate demand observations obtained from either a single or a limited number of sensing points in a building [Hart 1985; 1989; 1992, Sultanem 1991]. Sensing points are usually located at a building-level (e.g. main electricity meter), residence- or apartment-level (e.g. electric distribution board), or even for a small group of loads on an electrical socket-level (as in for example Gao et al. [2014]). NILM is a cost-effective, practical approach for monitoring end-use energy consumption at the single load level, in residential households, commercial buildings, and industrial facilities. It represents a compromise between distributed sensing and conditional demand analysis [Newsham and Donnelly 2013]. The former requires an expensive

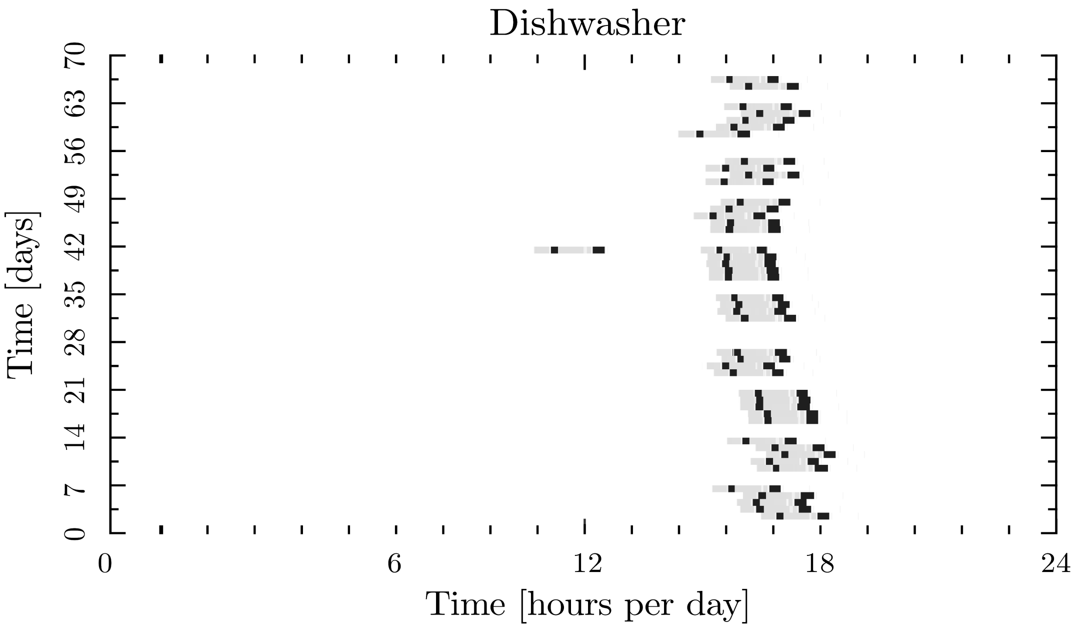
[1]Throughout this work, we use the two terms interchangeably with higher preference given to *energy disaggregation*. We also use the terms "load" and "appliance" interchangeably with higher preference given to the former.

and intrusive metering of individual appliances which is hardly feasible for large-scale population-wide deployments while the latter is mainly based on regression analysis of monthly bills and does not fulfill accuracy requirements for monitoring individual end-use loads.

Figure 1.1 depicts an example of a ten-week profile of a monitoring system for two loads from a commercial building, namely a dishwasher and a vending machine, projected on daily usage. The figure illustrates various key aspects of end-use energy monitoring. A main observation from the figure is the concept behind load identification and monitoring based on electrical load signatures. As observed from the figure, the two loads depict not only unique short-term power draw signatures but also distinct long-term usage patterns with one load being thermostatically controlled while the other being user activated. Additionally, the figure depicts two of the promising applications of energy monitoring system. The first is the usage abnormality of the dishwasher in the middle of the 42nd day, while the second is the malfunction of the vending machine leading to an energy leak for almost two days in the 7th week.

The first, and most focus of our work, is the promoted awareness of electrical energy usage especially for end-users. Energy disaggregation systems aim at providing customers with real-time, fine-grained feedback about their usage of electricity allowing them to adopt more efficient energy saving measures and strategies [Schleich et al. 2012]. In spite of the importance and adequacy of this objective, energy monitoring and disaggregation additionally offers a variety of promising applications, of which we discuss a few in the following.

Disaggregated end-use data can provide appliance manufacturers with detailed usage patterns of their products in real and long-lasting environments, allowing them to evaluate the effectiveness of their manufacturing innovations and identify weaknesses of their products. The motivation stems from the fact that newly manufactured appliances have witnessed significant improvements with respect to energy efficiency but a corresponding improvement within energy efficiency in buildings was not equally observed [Kavousian et al. 2015]. Detecting aging and malfunctioning loads that result in mysterious energy leakage (as illustrated in the example in Figure 1.1) is another application of monitoring systems leading end-users to educated decisions for their energy savings plans. Other promising applications of disaggregated data include occupancy detection, home automation [Patel et al. 2007],



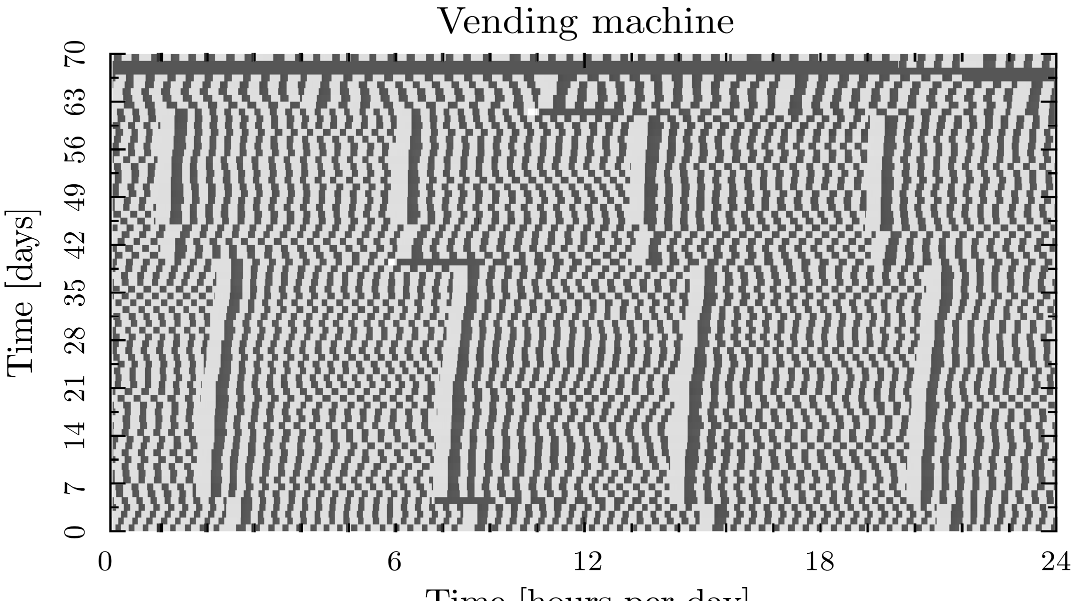


Figure 1.1.: Daily usage pattern of a dishwasher (upper) and a vending machine (lower) sub-metered from a commercial building in Berlin, Germany for 10 consecutive weeks. The figures illustrate the feasibility of inferring end-use loads merely from their patter of use. Gray-scale is proportional to the power draw of the load during operation.

itemized energy billing [Drenker and Kader 1999], and detection of unauthorized load usage [Cole and Albicki 1998].

In the last three decades, energy disaggregation has attracted a rapidly growing interest in many research groups worldwide. Energy disaggregation research occurred in three main directions in tandem, namely adapting more accurate disaggregation and inference algorithms, extracting more distinctive appliance signatures, and developing energy datasets that are suitable for training and evaluation of NILM systems. Reviews on various paradigms and approaches of end-use electrical energy disaggregation can be found in Zoha et al. [2012], Zeifman and Roth [2011], Froehlich et al. [2011], Jiang et al. [2011], Bonfigli et al. [2015] and Ruano et al. [2019], and we refer the interested reader to their publications for a detailed review.

A NILM system consists primarily of three sub-tasks, disaggregation, inference, and reconstruction. In the disaggregation phase, an aggregate power signal is decomposed into multiple components each of which corresponds to a single end-use load. In inference, each disaggregated component is assigned a load label (e.g. a load category, a specific load instance, or even anonymous identifier if labeled data is unavailable). Finally, load profile reconstruction aims at estimating the time series consumption patterns of disaggregated loads.

In this work, we address each NILM sub-task starting from load classification (in Chapter 2), and going though disaggregation and reconstruction either with available training data where deep data-driven models can be adopted (Chapter 5), or merely based on prior domain knowledge when labeled data is not provided (Chapter 3), or even approaches jointly leveraging both labeled and unlabeled data (Chapter 4).

1.2. Terminology Peculiars

We briefly give a remark on a few concepts the reader should be aware of in order to conceive our main focus and contribution in this work.

1.2.1. Intelligent or Smart ?

The cost-benefit analysis report by Ernst & Young for the Federal Ministry of Economics and Technology in Germany (BMWi) [Ernst & Young 2013, Bayer 2015] defined the two terms "*smart-*" and "*intelligent-metering*" differently. As stated by Bayer [2015], the report defined both terms as a digitization of the energy metering system in the sense that consumption profiles are not only sensed but also digitized, stored, and optionally communicated to interested parties. However, in intelligent metering architectures, real-time consumption profiles are only communicated to the end-users while in smart metering this information become additionally available to utility companies.

Based on this distinction, we advise the reader of this work to consider our contribution in the sense of intelligent metering, since our main focus in this work is the promoted awareness of end-users about their usage of energy.

1.2.2. Non-intrusive or Unsupervised ?

Many works on energy disaggregation tend to overload the concepts of supervised, unsupervised, and semi-supervised learning from the machine learning domain. Since this work proposes energy disaggregation models of all three learning paradigms, we first give two examples of these terminology peculiars followed by our adopted definitions of these three terms.

Parson et al. [2014] proposed a hidden Markov model (HMM) to build generalized load models from multiple instances that are then able to generalize to new load instances of the same category. Trained load models are then fine-tuned to specific instances in the target building for disaggregation based solely on labeled data. The authors indeed highlight the distinction between their approach and the machine learning definition of unsupervised training, yet they characterize their proposed model as unsupervised, since no labeling or training is required post-deployment. Such a model, from a machine learning perspective, would more precisely belong to the class of semi-supervised models since it trains from labeled measurements but is also permitted to leverage unlabeled data post-deployment.

As a second example, Cominola et al. [2017] proposed a two-stage load profile disaggregation and reconstruction framework comprised of a factorial hidden Markov model (fHMM) and an iterative dynamic time warping (DTW). Both stages required direct load measurements to construct a library of load signatures for training and disaggregation. The authors proposed to learn these signatures from the aggregate (that is, building-level) measurements if usage information is provided (referred to as *energy diaries* [Desmedt et al. 2009]). In spite of the savings in hardware costs, the proposed approach does not follow the machine learning definition of a semi-supervised learning model since unlabeled measurements are not leveraged for disaggregation. The authors simply utilize a novel, and claimed to be more convenient and less intrusive, labeling scheme.

In a nutshell, various researchers in the field of energy disaggregation adopt the terms *"un-* or *semi-supervised"* while they actually mean *"non-* or *semi-intrusive"* for installation and deployment in new target buildings. In our work, though, we adhere to the machine learning definitions of these concepts and, therefore, characterizing a system of being unsupervised necessitates that it does not required labeled or disaggregated training data neither post- nor pre-deployment. Additionally, a system is characterized of being supervised if its learning process is based merely on externally labeled data or semi-supervised if it leverages both labeled and unlabeled data.

1.3. Contributions

The main goal of this work is to explore a variety of end-use energy monitoring and disaggregation paradigms and propose feasible solutions, while investigating limitations, for each paradigm. In other words, we propose in this work energy disaggregation models for different levels and format of availability of training data, from entirely unavailable for which domain knowledge plays an important role, to abundantly attainable in which case data-driven deep models stand out. Our contributions in this work are:

1. Proposing and validating a data-driven model, namely, an ensemble of neural networks, for direct plug-level load monitoring (Chapter 2).

2. Building an event-based unsupervised electrical energy disaggregation system in which domain knowledge is the main reference for adjusting model hyper-parameters (Chapter 3).
3. Investigating the feasibility of semi-supervised event-based classification for load monitoring systems in situations where labeled data is available but scarce (Chapter 4).
4. Proposing an efficient, data-driven disaggregation model under the assumption of adequate availability of training data (Chapter 5).

1.4. Publications

- Karim Said Barsim, Bin Yang, "On the Feasibility of Generic Deep Disaggregation for Single-Load Extraction", *in the 4th International Workshop on Non-Intrusive Load Monitoring, March 2018, Austin, Texas* (best paper award).
- Karim Said Barsim, Lukas Mauch, and Bin Yang, "Neural Network Ensembles to Real-time Identification of Plug-level Appliance Measurements", *in the 3rd International Workshop on Non-Intrusive Load Monitoring (NILM 2016), Vancouver, Canada, 14th, May. 2016.*
- Karim Said Barsim and Bin Yang, "Sequential Clustering-based Event Detection for Non-Intrusive Load Monitoring", *in proceedings of the 6th International Conference on Computer Science and Information Technology (CCSIT 2016), Zurich, Switzerland, 2nd Jan. 2016.*
- Karim Said Barsim and Bin Yang, "Toward a Semi-Supervised Non-Intrusive Load Monitoring System for Event-based Energy Disaggregation", *in proceedings of the 3rd IEEE Global Conference on Signal and Information Processing (GlobalSIP), Orlando, Florida, USA, 14th, Dec. 2015.*
- Benjamin Wild, Karim Said Barsim, and Bin Yang, "A New Unsupervised Event Detector for Non-Intrusive Load Monitoring", *in proceedings of the 3rd IEEE Global Conference on Signal and Information Processing (GlobalSIP), Orlando, Florida, USA, 14th, Dec. 2015.*

- Karim Said Barsim, Roman Streubel, and Bin Yang, "Unsupervised Non- Intrusive Load Monitoring of Residential Appliances", *in proceedings of the 48th International Universities' Power Engineering Conference (UPEC), Dublin, 2nd-5th Sept. 2013.*

- Karim Said Barsim, Roman Streubel, and Bin Yang, "Unsupervised Adaptive Event Detection for Building-Level Energy Disaggregation", *in proceedings of the Power and Energy Student Summit (PESS), Stuttgart, 23rd-24th Jan. 2014.*

- Karim Said Barsim, Roman Streubel, and Bin Yang, "An Approach for Unsupervised Non-Intrusive Load Monitoring of Residential Appliances", *in proceedings of the 2nd Non-Intrusive Load Monitoring (NILM) Workshop 2014, Austin, 3rd Jun. 2014.*

1.5. Thesis Outline

We start this work by validating the feasibility of end-use load monitoring which serves additionally as an empirical estimation of the performance upper bound of an energy disaggregation system. Chapter 2 provides a case study on the problem of direct distributed sensing applied to a small set of residential loads, but across a wide range of deployment sites. The work introduces an ensemble of small artificial neural nets tasked all together with detection and classification of residential end-use load categories.

Electrical energy disaggregation (or NILM) starts from Chapter 3 in which we introduced a fully-unsupervised (in the machine learning sense) energy disaggregation system. The proposed system relays primarily on detecting abrupt changes in aggregate measurements, extracts features, clusters repeating events, and performs end-use energy estimation.

Chapter 4 relies on the assumption that labeled data may be provided but are expensive, and therefore scarce, and proposes a semi-supervised energy disaggregation system capable of leveraging both labeled and unlabeled observations in reaching better performance, even in the case of scarce labeled data. This chapter bridges the unsupervised disaggregation model introduced in the preceding chapter to the entirely data-driven model detailed in the following chapter.

In Chapter 5, we propose an end-to-end neural network architecture for training generic deep disaggregation systems from direct and aggregate load measurements. The model assumes no prior knowledge, and requires minimal external supervision for training beyond training data acquisition, and architecture selection.

Finally, we conclude this work in Chapter 6 with remarks on future work and development directions.

1.6. Notation

Throughout this work we primarily consider discrete-time signals denoted by $x(t)$ where $t \in \mathbb{N}$ is the independent time index starting from a reference epoch[2] $t = 1$. Should a signal have a physical interpretation, it is emphasized with a monospaced font as in the voltage signal $\mathtt{V}(t)$ or the reactive power signal $\mathtt{Q}(t)$. Vector-valued signals are noted with a boldfaced font as in $\boldsymbol{x}(t) \in \mathbb{R}^{d_x}$ which is a d_x-dimensional column vector for the time index t. By default, we use column vectors unless explicitly stated otherwise.

Finite sequences (oftentimes referred to as sub-sequences or segments) of the signal $x(t)$ over the interval $(a, b]$ where $b > a$ is denoted by

$$\boldsymbol{x}_{(a:b]} = \begin{bmatrix} x(a+1), & x(a+2), & \cdots, & x(b) \end{bmatrix}^\top \in \mathbb{R}^{(b-a)\times 1} \tag{1.1}$$

For example, $\boldsymbol{x}_{[a:b]}$ denotes the finite sequence $[x(a),\ x(a+1),\ \cdots,\ x(b)]^\top \in \mathbb{R}^{(b-a+1)}$. For brevity, $\boldsymbol{x}_{a:b} = \boldsymbol{x}_{(a:b]}$ will oftentimes be used as a shorthand for the half-open interval (that is, $(a : b]$) defined in Equation (1.1).

Analogously, a finite sequence of the discrete-time d_x-dimensional signal $\boldsymbol{x}(t)$ over the interval $(a, b]$ is given by the matrix $\mathbf{X}_{(a:b]}$ where

$$\mathbf{X}_{(a:b]} = \begin{bmatrix} \boldsymbol{x}(a+1), & \boldsymbol{x}(a+2), & \cdots, & \boldsymbol{x}(b) \end{bmatrix}^\top \in \mathbb{R}^{(b-a)\times d_x} \tag{1.2}$$

[2]That is to say that natural numbers in this work are defined as the set of positive integers.

and likewise for $\mathbf{X}_{(a:b)}$, $\mathbf{X}_{[a:b)}$, and $\mathbf{X}_{[a:b]}$. Particularly for vector-valued signals $\boldsymbol{x}(t)$ the denotation $\boldsymbol{x}_{(a:b]}$ is defined as

$$\boldsymbol{x}_{(a:b]} = \mathrm{vec}\big(\mathbf{X}_{(a:b]}\big) \in \mathbb{R}^{(b-a)d_x \times 1} \tag{1.3}$$

where $\mathrm{vec}(\,\cdot\,)$ is the matrix vectorization operator and $\mathbf{X}_{(a:b]}$ is as defined in Equation (1.2). Similar to scalar-valued signals, the shorthand notation $\mathbf{X}_{a:b} \equiv \mathbf{X}_{(a:b]}$ and $\boldsymbol{x}_{a:b} \equiv \boldsymbol{x}_{(a:b]}$ will refer to Equation (1.2) and Equation (1.3), respectively. In fact, the sub-sequence definition of scalar-valued signal given in Equation (1.1) can be seen as a special case of the vectorization given by Equation (1.3) for vectorial signals.

In addition, we will repeatedly make use of the translational operator $\mathfrak{T}_\tau(\,\cdot\,)$ defined on the signal $\boldsymbol{x}(t)$ as

$$\mathfrak{T}_\tau \boldsymbol{x}(t) = \boldsymbol{x}(t+\tau) \tag{1.4}$$

and on the sequence $\boldsymbol{x}_{a:b}$ as

$$\mathfrak{T}_\tau \boldsymbol{x}_{a:b} = \boldsymbol{x}_{(a+\tau):(b+\tau)} \tag{1.5}$$

and its inverse $\mathfrak{T}_\tau^{-1}(\,\cdot\,)$ defined as

$$\mathfrak{T}_\tau^{-1} \boldsymbol{x}(t) = \boldsymbol{x}(t-\tau) \tag{1.6}$$

on the signal $\boldsymbol{x}(t)$ and

$$\mathfrak{T}_\tau^{-1} \boldsymbol{x}_{a:b} = \boldsymbol{x}_{(a-\tau):(b-\tau)} \tag{1.7}$$

on the sequence $\boldsymbol{x}_{a:b}$. Equation (1.6) implies a signal delay (i.e. time-shift) by τ.

We do not differentiate between random variables and realizations thereof so that $p(x) = p(x = x)$ both equally represent the probability that the random variable x is assigned the value x. To further avoid pedantry, we let $p(x)$ denote the $\underline{\text{p}}$robability $\underline{\text{d}}$ensity $\underline{\text{f}}$unction (PDF) of the random variable x if x is continuous-valued and the $\underline{\text{p}}$robability $\underline{\text{m}}$ass $\underline{\text{f}}$unction (PMF) when x is discrete-valued. Each case will be clearly obvious from the context.

The subscripted real number set $\mathbb{R}_{\geqslant a}$ denotes the half-bounded interval $[a, \infty)$ and in a similar fashion for $\mathbb{R}_{>a}$, $\mathbb{R}_{\leqslant a}$, and $\mathbb{R}_{<a}$. Likewise, the subscripted integer number set $\mathbb{Z}_{\geqslant a}$ denotes the integer sequence $a, a+1, \ldots$ and similarly for $\mathbb{Z}_{>a}$, $\mathbb{Z}_{\leqslant a}$, and $\mathbb{Z}_{<a}$.

Finally, we point out that exceptions to these rules may arise if deemed necessary but shall be explicitly noted.

Chapter 2.

Load Identification under Plug-Level Monitoring

Many works on energy disaggregation are motivated by their superiority over direct distributed sensing approaches. That is in the sense that, the former is cost- and energy-efficient for scalable deployments, less invasive to consumers' properties and privacy, and more capable of leveraging existing architectures and the recent smart meter rollouts (amongst many other motivations but to mention just a few), while expected to maintain the accuracy and monitoring performance of a distributed sensing deployment.

From this view point, direct sensing is not only considered the other extreme of greedy intrusive monitoring but also the upper bound with respect to monitoring performance in spite of being invasive or infeasible for large scale deployments. Such a view, however, necessitates an estimation of the performance of direct distributed sensing deployments viewed as an upper bound on the performance of energy disaggregation methods and treated as an assessment of one of its late stages namely, load identification.

This chapter is organized as follows. We first introduce the problem of load identification in Section 2.1 as a common problem to both energy disaggregation and distributed sensing systems followed by a brief review of prior art on this problem in Section 2.2. The addressed challenge is further explicitly formulated in Section 2.3. Our contributions then follow starting from Section 2.4 with a detailed description of the adopted raw voltage and current waveforms as load signatures. In Section 2.5, we detail our proposed robust real-time identification approach based on an ensemble of neural networks. Performance assessment measures for the adopted model are introduced in Section 2.6. Afterwards, we report and discuss empirical evaluations on a real-word plug-level dataset [Gao et al. 2015] obtained from a variety of common end-use load categories and across different households in Section 2.7. Finally, Section 2.8 concludes this chapter.

Work in this chapter has been published by Barsim and Yang [2015] and formulations, phrasing, experiments, and results are partially adopted from our prior work.

2.1. Introduction

We define the task of *load identification* as the process of inferring abstract load information from low-level load usage data. Complexity of such a task depends on the extent to which load information is to be inferred and the level of granularity to which load usage data is provided. For example, load usage data can be provided in terms of weakly/monthly usage information (e.g. number of hours of usage per day/weak), *on-off* timing patterns, load energy profiles, or raw current and voltage measurements. The targeted abstract load information ranges from generic load characterization such as operational principles (e.g. resistive, thermostatically-controlled, pump-operated, motor-driven, switching mode power supplied (SMPS) [Gupta et al. 2010], ... etc) [Sultanem 1991], functional aspects (e.g. temperature controller, lighting, user interactive, ... etc), miscellaneous categorization (e.g. air conditioner, heater, lamp, computer, printer, ... etc), and even down to specific load instances in an electrical network (e.g. a certain lighting circuit in an apartment) as in [Gupta et al. 2010].

Direct distributed sensing is similarly motivated by all promising applications of energy disaggregation (e.g. home automation, itemized billing, health monitoring, elderly assistance, demand response, energy efficiency, promoted self-awareness of energy usage, ... etc). However, superiority of energy disaggregation stems from the scalability barrier faced by distributed sensing architectures where large scale[1]

[1]That is, population-wide deployments rather than single user cases which have been successfully realized as in [Kelly and Knottenbelt 2015b].

deployments are normally either prohibitively infeasible, expensive, or even energy inefficient.

2.2. Related Work

In this section, we briefly review two parallel tracks of prior art. The first track utilized the high-frequency instantaneous voltage and current trajectories (referred to hereafter as VI-trajectories and will be formally defined in Section 2.4) or engineered features thereof as a load signature. The second track addresses the problem of load identification via distributed plug-level monitoring.

Lam et al. [2007] utilized steady-state VI-trajectories to blindly construct a taxonomy of load signatures mainly based on the residential load categorization proposed by Sultanem [1991]. The authors extracted a predefined set of geometrical features from each normalized VI-trajectory for a hierarchical clustering algorithm [Aggarwal and Reddy 2013] to construct the targeted taxonomy. They then showed that these features proved to be more effective in classifying residential loads than traditional power measures (such as root mean square (RMS) current, power factor, ... etc) while pertaining a similar level of interpretability.

Gao et al. [2014] published the plug-level appliance identification dataset[2] (PLAID) followed by an extended version by Baets et al. [2017b]. PLAID comprises direct voltage and current measurements of different household loads at the relatively high sampling frequency of 30 kHz with the standard grid frequency of 60 Hz. In their dataset, end-use loads were grouped into 11 categories, spread over 55 residential buildings, and with an average of almost 100 load instances per category. PLAID will be the target evaluation dataset in this chapter and will be described in more detail Section 2.7.1. The main purpose of this dataset is to conduct a proper generalizability assessment of proposed load identification algorithms to new residential buildings.

Gao et al. [2015] reported, shortly afterwards, with a feasibility study on load identification empirically validated on the PLAID dataset. In their study, Gao et al. trained and compared different off-the-shelf classifiers (random forests, support

[2]Found under: `http://plaidplug.com`

vector machines, naïve Bayes, nearest neighbors, amongst others) on the binary-valued images constructed from the the VI-trajectories (referred to in their work as VI-binary-images). The authors reported benchmarking identification metrics on PLAID households, and additionally emphasized the improved identification accuracy due to their proposed features compared to either raw measurements, power signals, or other engineered features. Reported identification accuracies reached up to 86% of total identification queries based on a leave-one-(building)-out cross validation[3] using random forest trees (RFTs) and a combined set of features in addition to raw measurements as load signatures. The authors additionally reported the notable degradation of identification accuracy with a reduction in sampling rates for almost all adopted models, and the challenging implementation on embedded systems due to the complexity of adopted features. We address both limitations in this chapter, and report a new load identification benchmark using an ensemble of small neural nets.

De Baets et al. [2018] applied convolutional neural networks (CNN) on a variant of VI-image features (with non-binary pixel values) and RFTs on VI-shape descriptors (known as elliptic Fourier features) [Baets et al. 2017a, Kuhl and Giardina 1982] and evaluated on an extended version of PLAID [Baets et al. 2017b] and the worldwide household and industry transient energy dataset (WHITED) [Kahl et al. 2016].

Following our work Barsim et al. [2016], Baptista et al. [2018] and Davies et al. [2019] reported on the application of CNN architectures for load identification validated on PLAID. They emphasized the advantage of the translation-equivariant temporal convolutional layers and the multi-class architectures that notably reduce training time. Undoubtedly, depth in hierarchical architectures increases representative power of a neural network model. Also, connectivity patterns can indeed encode domain knowledge when carefully designed. Distinct from these works, however, our proposed model features two main advantages. The first is the modularity in the adopted one-vs-one strategy. Such modularity permits ease of domain knowledge incorporation (e.g. knowing a priori that a specific target building does not have an certain load category) with a notable reduction in model size compared to multi-class networks. The second is the simple, relatively shallow architecture fea-

[3]We adopted the same cross validation scheme and shall, therefore, detail their approach explicitly in Section 2.7.

turing low computational complexity suitable for embedded system deployments and real-time monitoring.

2.3. Problem Statement

The problem we address in this chapter is formally stated as follows. *Estimate the identifiability of end-use household loads from direct instantaneous voltage and current measurements*. This formulation is further detailed in the sequel.

Identifiability in the aforementioned formulation is addressed to the level of an adequate categorization of end-use loads, and an explicit example of which shall be given in Section 2.7.

The term "*direct measurements*" implies single-load circuits (that is, monitoring a single load a time). Direct measurements can be obtained from in-series current sensors attached to the plug of a load, and hence the name "*plug-level monitoring*". This assumption mimicks the behavior of an ideal energy disaggregation system where a disaggregator has successfully reconstructed end-use load profiles of each target load and the task at this stage is to classify each disaggregated signature to one of the end-use categories.

Acquisition of the *instantaneous* voltage and current signals implicitly implies a sampling rate higher (oftentimes orders in magnitude higher [Carrie Armel et al. 2013]) than the grid frequency as shall become more evident in the following section.

We additionally investigate further aspects closely related to the targeted problem. Examples of these aspects include the effect of the sampling frequency (Section 2.7.6), size of training data (Section 2.7.5), and external knowledge (Section 2.7.4) on the identifiability of end-use loads from their direct voltage and current measurements.

2.4. Load Signatures

In this work, we utilize the *raw* voltage $\texttt{v}(t)$ and current $\texttt{i}(t)$ waveforms (henceforth referred to as VI-waveforms) during the steady-state operation of a load as its distinct signature, where $t \in \mathbb{N}$ represents a time index. Such a signature is extracted from multi-state and variable loads[4] randomly while operating in any of their *on*-states.

Presumably, VI-waveforms are the most informative electricity-related features since they completely characterize an electric circuit, in the sense that all electricity-related load signatures[5] are derived quantities from these raw instantaneous waveforms.

In order to illustrate our motivation for adopting raw VI-waveforms as load signatures, we first illustrate their behavior in elementary circuits and compare them with what is commonly referred to as VI-trajectories.

2.4.1. Identifiability in Elementary Circuits

Figure 2.1 shows a generic diagram of an alternating-current (AC) circuit. For ease of illustration, we will limit the discussion to ideal AC voltage sources with a purely sinusoidal waveform, and a linear time-invariant (LTI) load[6]. In this simple model, the load's opposition to the supply voltage and the resulting current is characterized by its total electrical impedance.

Denote the impedance of a load by $\mathtt{Z} \in \mathbb{C}$ which is a complex-valued quantity whose real part is referred to as the resistance $\mathtt{R} \in \mathbb{R}_{\geqslant 0}$ (measured in Ohm), where $\mathbb{R}_{\geqslant a}$ denotes the interval $[a, \infty)$, and the imaginary part is referred to as the reactance $\mathtt{X} \in \mathbb{R}$ so that $\mathtt{Z} = \mathtt{R} + j\mathtt{X}$ where j is the imaginary unit. Equivalently, let $\mathtt{Z} = |\,\mathtt{Z}\,| \exp(\phi_{\mathtt{Z}})$ where $|\,\mathtt{Z}\,| = \sqrt{\mathtt{R}^2 + \mathtt{X}^2}$ is the amplitude of the load impedance $\mathtt{Z}$ and $\phi_{\mathtt{Z}} = \tan^{-1}(\mathtt{X}/\mathtt{R})$ is its phase.

[4]Hart [1992] proposed splitting end-use loads into three model classes: 1) an *on-off* load (e.g. a lamp or an iron) which operates under a single *on*-state with nearly stationary power draw, 2) a multi-state load, named in the original text as a "finite state machine", which has multiple *on*-states but each of which features a nearly stationary power draw, and 3) continuously variable loads with non-stationary power draw in general such as power tools.

[5]Energy disaggregation can also benefit from non-electricity-related features such as weather conditions [Barker et al. 2012], electromagnetic detectors [Patel et al. 2007, Gupta et al. 2010, Hazas et al. 2011], or portable and wearable devices' signals [Roy et al. 2016]. Such features are outside the scope of this work.

[6]An example of which are all loads composed of passive LTI circuit components such as resistors, capacitors, and inductors. This is definitely a strong assumption since most modern household loads are equipped with complex non-linear electronic components. Nevertheless, non-linearities will most likely promote identifiability as shall be illustrated in Figure 2.3 but rather tackle reliable feature extraction especially at low sampling rates and under noisy measurements.

Let $\mathtt{v}(t)$ denote the discrete-time real-valued sinusoidal waveform of the supply voltage sampled at F_s such that $\mathtt{v}(t) = |\,\mathtt{V}\,| \cos(2\pi f t+\phi_{\mathtt{V}})$ or equivalently the real part of its analytic form representation $\Re\{\mathtt{v}^c(t) = |\,\mathtt{V}\,| \exp(j(2\pi f t+\phi_{\mathtt{V}}))\}$ where $|\,\mathtt{V}\,| \in \mathbb{R}_{\geqslant 0}$ is the voltage amplitude, $\phi_{\mathtt{V}} \in [-\pi/2,\, \pi/2]$ is the phase-shift of the voltage signal, $\Re\{\cdot\}$ is the real part denotation, $\exp(\cdot)$ is the complex exponential function, and f is the normalized frequency defined as $f = F_g/F_s$ where F_g is the grid frequency[7] and F_s is the sampling rate. The assumption of linearity permits re-writing Ohm's law in the complex representation

$$\mathtt{v}^c(t) = \mathtt{Z} \cdot \mathtt{i}^c(t) \tag{2.1}$$

for any time index t where $\mathtt{i}^c(t)$ is the analytic representation of the current signal $\mathtt{i}(t)$ at the time index t. It becomes evident that the resulting current signal $\mathtt{i}(t) = \Re\{\mathtt{i}^c(t)\}$ is of a sinusoidal waveform whose amplitude $|\,\mathtt{I}\,| \in \mathbb{R}_{\geqslant 0}$ is given by $|\,\mathtt{V}\,| / |\,\mathtt{Z}\,|$ whereas its phase $\phi_{\mathtt{i}} \in [-\pi/2,\, \pi/2]$ is shifted[8] by $\phi_{\mathtt{Z}}$ from the voltage signal so that $\mathtt{i}(t) = \Re\{(|\,\mathtt{V}\,| / |\,\mathtt{Z}\,|) \exp(j(2\pi f t + \phi_{\mathtt{V}} - \phi_{\mathtt{Z}}))\}$. The supply voltage $|\,\mathtt{V}\,|$ and the grid frequency F_g are country specific power system standards and are either ~160 or ~330 voltage (in amplitude, that is, 110-120 or 220-240 in RMS) for the former, and 50 or 60 Hz for the latter.

The VI-trajectory of a load is the directed path in $\mathbb{R}^2$ constructed by the mapping $\boldsymbol{\rho} : \mathbb{N} \to \mathbb{R}^2$ defined as $\boldsymbol{\rho}(t) = [\mathtt{v}(t),\ \mathtt{i}(t)]^\top$ where $[\,\cdot\,]^\top$ is the matrix transpose denotation. In the case of periodic signals (as in an AC-circuit with an LTI-load) and assuming rationality of the normalized frequency $f \in \mathbb{Q}$, then the VI-trajectory forms a loop such that $\boldsymbol{\rho}(t) = \boldsymbol{\rho}(t + n)$ for some positive integer $n \in \mathbb{N}$.

Finally, we refer to the signals $\tilde{\mathtt{v}}(t) = \mathtt{v}(t)/\,|\,\mathtt{V}\,|$ and $\tilde{\mathtt{i}}(t) = \mathtt{i}(t)/\,|\,\mathtt{I}\,|$ and the mapping $\tilde{\boldsymbol{\rho}}(t) = [\tilde{\mathtt{v}}(t),\ \tilde{\mathtt{i}}(t)]^\top$ as the *normalized* voltage, current, and VI-trajectory, respectively, where amplitude information has been discarded.

We illustrate in Figure 2.2 the behavior of the voltage and current signals along with their VI-trajectories for simple examples of LTI loads such as purely resistive loads (R-loads), series resistive-capacitive loads (RC-loads), and series resistive-inductive loads (RL-loads).

[7]Also known as the fundamental-, line-, or utility-frequency.
[8]Either leading or lagging the voltage signal based on the direction of the phase shift.

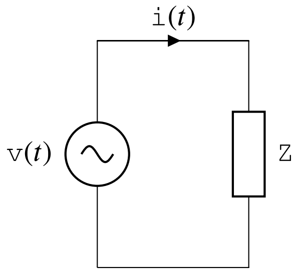


Figure 2.1.: A generic LTI load under a sinusoidal supply voltage $v(t)$ and the resulting current $i(t)$ where Z denotes the total electrical impedance of the load.

In a purely resistive load (top row in Figure 2.2) where the impedance is real-valued $Z = R$, both the voltage $v(t)$ and current $i(t)$ signals remain in phase, meaning that $\phi_i = \phi_v$. The resulting VI-trajectory $\boldsymbol{\rho} = [v(t), (v(t)/R)]^\top$ in this case follows a straight line passing through the origin whose slope is $\frac{1}{R}$.

The impedance of a series RC-load (middle row in Figure 2.2) is given by $R - j/(2\pi F_g C)$ where $C \in \mathbb{R}_{>0}$ is the capacitance (measured in Farads). On the other hand, a series RL-load (lower row in Figure 2.2) has an impedance of $R + j(2\pi F_g L)$ where $L \in \mathbb{R}_{>0}$ is the inductance value (measured in Henries). Accordingly, the impedance phase ϕ_Z is positive for RL-circuits, and negative for RC-circuits.

The middle and lower rows of Figure 2.2 illustrate the behavior of RC- and RL-circuits, respectively. In this case, the current leads or lags the voltage signal, respectively, with a phase difference equivalent to the negative impedance phase ϕ_Z. The VI-trajectories in these cases are tilted elliptic loops directed clock-wise in the RC-circuit, and counter clock-wise in the case of an RL-circuit. In the extreme case of a purely capacitive or inductive circuit, the total impedance degenerates to a purely imaginary reactance $Z = jX$ with a phase value $\phi_Z = \pm\pi/2$ rendering the voltage and current signals mutually orthogonal, and resulting in a circular VI-trajectory looping clock-wise or counter clock-wise in the capacitive and inductive circuits, respectively.

In general, certain load characteristics can be inferred from geometrical features (e.g. enclosed area, major axis slope, looping direction ... etc) of the VI-trajectory. Such an approach has been adopted by previous works as in [Lam et al. 2007] and [Hassan et al. 2014] to name a few examples.

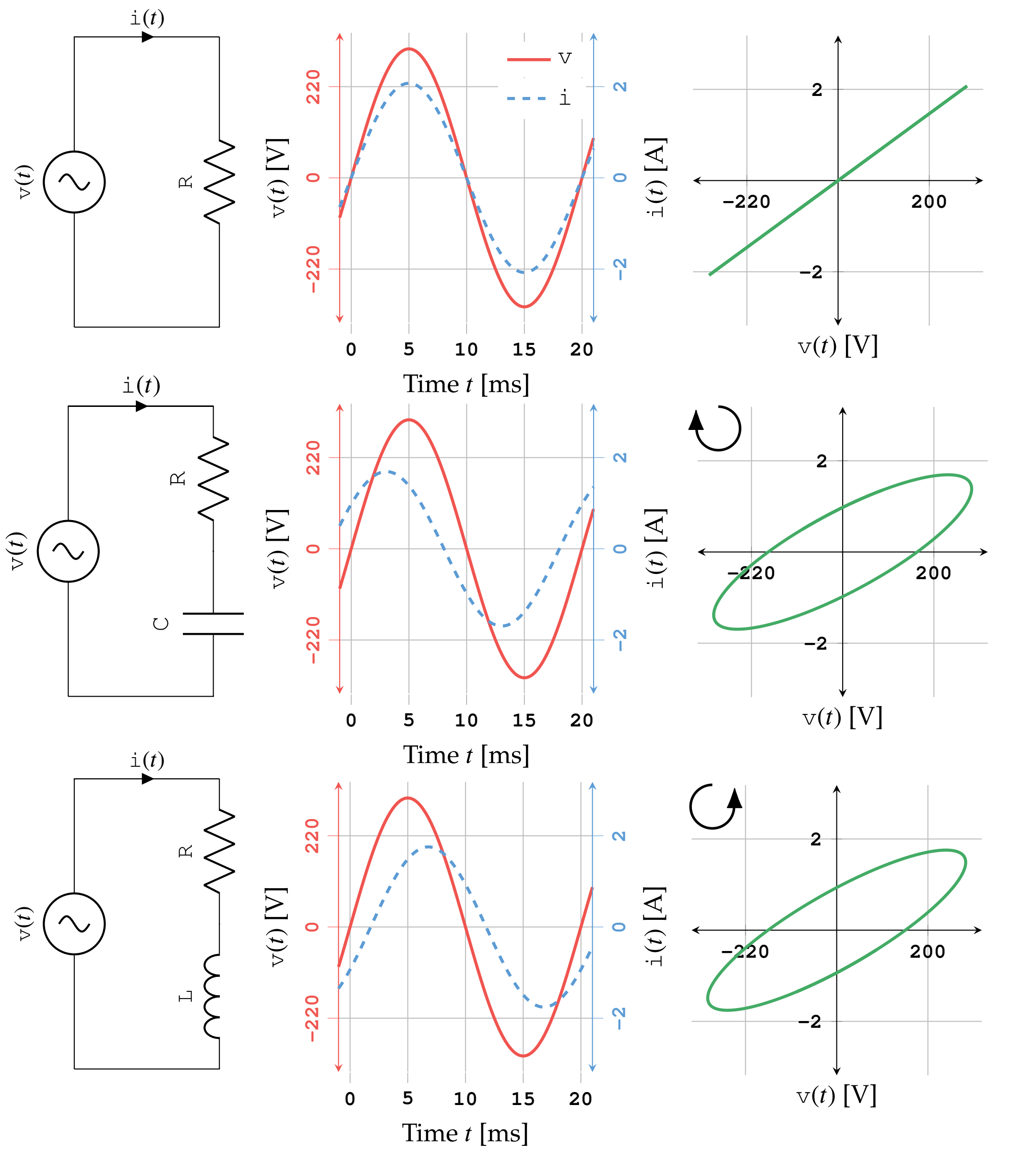


Figure 2.2.: Examples of elementary LTI load circuits. A purely resistive load (top), an RC-load (middle), and an RL-load (bottom). Rotational curvature of the rightmost VI-trajectories are indicated by the upper left circular arrow. Simulations are based on the selected values $|\mathtt{V}| = 220\sqrt{2}$ V, $\phi_{\mathtt{V}} = \pi/2$, $F_g = 50$ Hz, $\mathtt{R} = 150\ \Omega$, $\mathtt{C} = 30\ \mu$F, and $\mathtt{L} = 300$ mH which are adopted for illustration purposes only.

As load non-linearities[9] are introduced, deviations from the ideal elliptically shaped VI-trajectories will be observed. Examples of these deviations include non-elliptic loops, self intersections, and peaks introduced in the outer (i.e. leftmost and rightmost) segments of the trajectory.

Figure 2.3 depicts samples of normalized VI-trajectories (middle column) of variants of household loads extracted from the PLAID dataset [Gao et al. 2014] along with the normalized raw voltage $\tilde{\mathtt{v}}(t)$ (right) and current $\tilde{\mathtt{i}}(t)$ (left) signals. Loads shown are an $\underline{\text{a}}$ir $\underline{\text{c}}$onditioner (AC), a $\underline{\text{c}}$ompact $\underline{\text{f}}$luorescent $\underline{\text{l}}$amp (CFL), a fan, $\underline{\text{fr}}$idge (FR), a hairdryer, a heater, a light bulb, a laptop, a $\underline{\text{mic}}$rowave-oven, a $\underline{\text{v}}$acuum $\underline{\text{c}}$leaner (VC), and a $\underline{\text{w}}$ashing $\underline{\text{m}}$achine (WM), in that order from top downwards in the figure. As expected, non-linear loads (e.g. a laptop or a CFL) show a clear deviation from the ideal elliptic VI-trajectory

These deviations require more carefully engineered geometrical features to reach a targeted level of identifiability [Lam et al. 2007, Hassan et al. 2014]. With increasing non-linearities and noisy measurements, it rapidly becomes infeasible to design reliable, adequately generic signatures based on the VI-trajectories. This is one of our key motivations for adopting raw VI-waveforms along with a data-driven model that comprises unsupervised feature extraction capabilities as shall be discussed in Section 2.5.

A notable observation from Figure 2.2 is that loosing the directionality of the VI-trajectories renders some loads unidentifiable as illustrated in the hypothetical case of RC- and RL-circuits of the figure where both lead to the same loop just oppositely directed. This observation is another key difference between our contribution and prior art that leveraged unsupervised feature extraction on VI-trajectories[10] [Gao et al. 2015] where directionality is lost resulting in a reduced level of identifiability. With the adopted VI-waveforms (formally defined in the following section) as load signatures, we claim to maintain a higher level of identifiability and empirically show superior performance on the benchmarking dataset.

[9]Non-linear loads are those comprising non-linear circuit components such as diodes and transistors which are the major components in nowadays complex power electronics or digital logic circuits.

[10]In fact, Gao et al. [2015] derived a VI-image feature which is a mesh of pixels constructed from the VI-trajectory. Nevertheless, directionality is still lost leading to inferior identifiability.

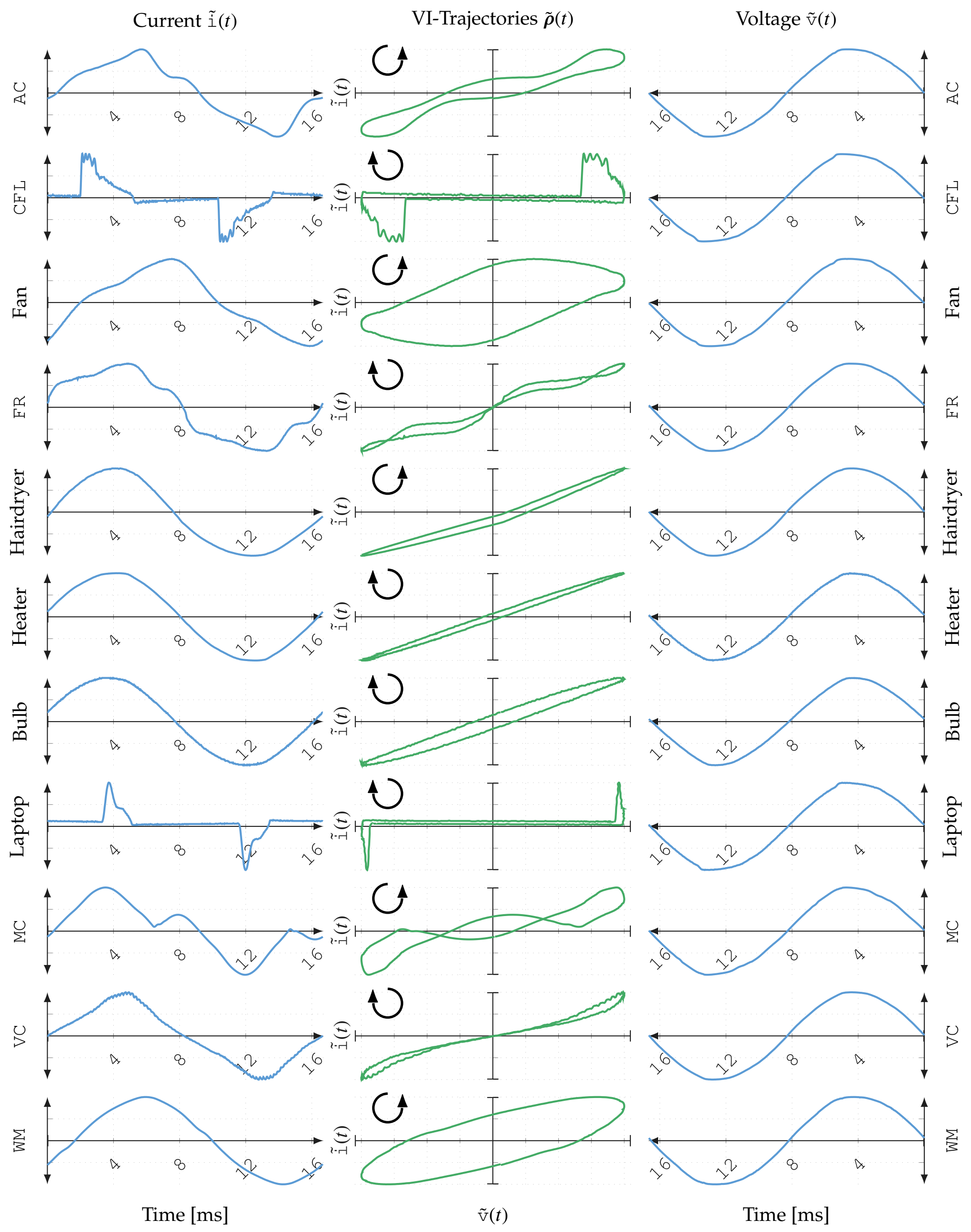


Figure 2.3.: Normalized voltage $\tilde{v}(t)$ (right), current $\tilde{i}(t)$ (left), and VI-trajectories $\tilde{\rho}(t)$ (middle) of selected samples from each load category in PLAID [Gao et al. 2014]. Samples are shown for the very last period of each measurement. Abbreviated load names are found in text.

2.4.2. VI-Waveforms as Load Signatures

As mentioned earlier, we adopt the *raw* VI-waveforms as load signatures and exploit one of the learning capabilities of neural nets as data-driven unsupervised feature extractors. In this subsection, we formalize the definition of the VI-waveform and defend our motivation for adopting this generic load signature.

We first remind the reader of our notation for a sub-segment of a time series signal (see Section 1.6). The vectorized a-to-b sub-segment $[\boldsymbol{x}(a+1), \boldsymbol{x}(a+2), \dots, \boldsymbol{x}(b-1), \boldsymbol{x}(b)]^\top$ of a d_x-dimensional discrete-time signal $\boldsymbol{x}(t)$ is denoted by $\boldsymbol{x}_{a:b} \in \mathbb{R}^{(b-a)d_x \times 1}$ where $a, b \in \mathbb{N}$ and $b > a$ so that

$$\boldsymbol{x}_{a:b} = \mathrm{vec}\Big(\big[\boldsymbol{x}(a+1), \boldsymbol{x}(a+2), \dots, \boldsymbol{x}(b-1), \boldsymbol{x}(b)\big]^\top\Big) \in \mathbb{R}^{(b-a)d_x \times 1} \tag{2.2}$$

where $\mathrm{vec}(\cdot)$ is the matrix vectorization operator. By adopting this notation, we then define the VI-waveform.

Let the signal $\boldsymbol{x}(t)$ denote the concatenation of the normalized current $\tilde{\mathrm{i}}(t)$ and voltage $\tilde{\mathrm{v}}(t)$ signals at time index t such that[11]

$$\boldsymbol{x}(t) = \begin{bmatrix} \tilde{\mathrm{i}}(t) \\ \tilde{\mathrm{v}}(t) \end{bmatrix} \quad \in \mathbb{R}^{2\times 1} \tag{2.3}$$

The VI-waveform is, accordingly, defined as the d-historical samples of the current and voltage signals till time index t (inclusive) and will be hereafter denoted by

[11]The range of the normalized signals is in fact $[-1, 1]^{2\times 1}$ but we rather keep the formulation more generic to accommodate, for instance, outlier-aware normalization.

$\boldsymbol{x}_{t-d:t}$

$$\boldsymbol{x}_{t-d:t} = \begin{bmatrix} \tilde{\mathtt{i}}_{t-d:d} \\ \tilde{\mathtt{v}}_{t-d:d} \end{bmatrix} = \begin{bmatrix} \tilde{\mathtt{i}}(t-d+1) \\ \tilde{\mathtt{i}}(t-d+2) \\ \vdots \\ \tilde{\mathtt{i}}(t-1) \\ \tilde{\mathtt{i}}(t) \\ \tilde{\mathtt{v}}(t-d+1) \\ \tilde{\mathtt{v}}(t-d+2) \\ \vdots \\ \tilde{\mathtt{v}}(t-1) \\ \tilde{\mathtt{v}}(t) \end{bmatrix} \in \mathbb{R}^{2d\times 1} \tag{2.4}$$

where d is a design hyperparameter and is chosen in all our experiments such that the VI-waveform comprises one complete period of the voltage and current signals based on the grid- and sampling-frequencies. In other words,

$$d = \left\lceil \frac{F_s}{F_g} \right\rceil \tag{2.5}$$

and accordingly ranges in our experiments from 42 samples for the 2.5 kHz sampling frequency to 500 samples for the 30 kHz original sampling frequency of PLAID, where $\lceil\cdot\rceil$ denotes the ceiling function. The VI-waveform $\boldsymbol{x}_{t-d:t}$ is the input signal to all our models.

Our choice of a load signature along with unsupervised feature extractors are motivated by two main advantages over prior work. First, our approach permits robust adaptation to different sampling rates and noisy measurements for reliable extraction of stable representative features. Geometric features of VI-trajectories (as adopted in [Hassan et al. 2014, Gao et al. 2015, Lam et al. 2007]) require an educated guess from domain experts for generic, representative set of features. Yet, they remain vulnerable to measurement noise and discretization error at lower sampling rates.

Second, raw VI-waveforms preserve valuable temporal information (such as the directionality of the phase shift) that in some cases accommodate the most (or even the only) discriminative feature between certain loads. An example of such a case has been discussed in the preceding section and illustrated in the RC- and RL-circuits of Figure 2.2. Undirected VI-trajectory features (e.g. VI-images [Gao et al. 2015]) discard this information and are, accordingly, expected to suffer from unidentifiability limitations.

We, however, support the repeatedly reported claim that amplitude information is to be discarded in order to support generic load identification (i.e. a higher level of abstraction than specific instance identification) and avoid calibration problems. Therefore, we discard all amplitude information through normalization similar to [Gao et al. 2015].

2.4.3. Training Data

Let $\mathcal{D}^{(\text{train})}$, or simply $\mathcal{D}$ to avoid tedious pedantry, denote the set of training data comprised of N pairs of load signatures $\boldsymbol{x}_{t-d:t}$ and the corresponding load category y such that

$$\mathcal{D} = \left\{ \left(\boldsymbol{x}^{(n)}_{t-d:t}, y^{(n)}\right) \right\}_{n=1}^{N} \tag{2.6}$$

where the superscript n denotes the nth data item, N is the size of the dataset, $\boldsymbol{x}^{(n)}_{t-d:t}$ is the VI-waveform (signature) of the nth load, and $y^{(n)} \in \mathbb{Y}$ is the corresponding load category with $\mathbb{Y} = \{y_m\}_{m=1}^{M}$ being the space of M load categories. In $\mathcal{D}$, and without yet any sort of data augmentation (which is to be discussed next) the load signature $\boldsymbol{x}^{(n)}_{t-d:t}$ is the very last period of each load measurement in PLAID.

For notational convenience, we will occasionally omit the superscript $\cdot^{(n)}$ unless this becomes a source of confusion.

2.4.4. Data Augmentation for Translational Invariance

In the preceding subsections, the definition of the load signature $\boldsymbol{x}_{t-d:t}$ comprised an unspecified parameter, namely, the time index t. The time index t is an indicator of the phase of the voltage and current signals, and a change in that index maps to a phase shift (or a translation) in each signal.

In unsupervised data-driven feature extractors (such as the model adopted in this work), it is of a paramount importance to avoid leaking information about the target variable (the category of the load in our case) through irrelevant variables (e.g. the time index). For example, with an unlucky selection of the sub-segments' phase per load the model may learn to predict the load category merely from the phase (or time index) of the corresponding sub-segment rending itself invaluable for meaningful predictions.

It is, therefore, essential to attain non-informativeness of the irrelevant time index t with regard to the target variable (load category) in order to achieve reliable feature extraction. Explicitly stated, it is required to render the model invariant to translations in the input signals. While there exist neural net models that enable encoding such invariances directly in their architectures, we rather adopt an alternative and commonly adopted approach to achieve translation invariance known as *data augmentation* or *algorithmic expansion of training data*.

Formally stated , the objective is to render the time index variable t non-informative with regard to the load category y such that

$$p(y \mid t) = p(y) \tag{2.7}$$

where $p(y \mid t)$ is the conditional probability of the load category y given the time index t while $p(y)$ is the prior distribution of load categories (interpreted, for instance, as how often a certain load is found in addressed buildings). In this case, the two random variables y and t are said to be statistically independent and this property is denoted by $y \perp\!\!\!\perp t$. Let $p(y, t)$ denote the joint distribution of both variables, then it holds from Equation (2.7) that

$$\begin{aligned} p(y, t) &= p(y \mid t)\, p(t) \\ &= p(y)\, p(t) \end{aligned}$$

but also

$$p(y, t) = p(t \mid y)\, p(y)$$

from which it holds

$$p(t \mid y) = p(t) \tag{2.8}$$

(which also serves as the proof of the symmetry property of statistical independence). Relevant to this discussion though is the implication of Equation (2.8) which is stated as follows. The distribution of the phase shift is identical over all load categories. This requirement can be satisfied by either aligning phases[12] of all measurements such that t follows a delta distribution[13] or randomly selecting the phase of each measurement so that t follows a uniform distribution over its support. The former approach is vulnerable to noisy measurements and alignment artifacts. The latter requires a relatively large dataset (free of biases) to achieve uniformity. For the example of PLAID with around 100 measurements per category and 500 bins of support for the phase shift t, neither uniformity nor independence (condition Equation (2.8)) is likely to take place.

In order to satisfy Equation (2.8), the training dataset is augmented with sets of translated sub-segments for each measurement such that the phase is always uniformly distributed conditioned on any measurement. Stated differently, all loads contribute equally to the augmented training set with respect to phase shift, so that the latter is no longer informative about from which load it has been produced. We first formulate the translational invariance data augmentation, followed by a practical consideration.

Let $\mathfrak{T}\{\cdot\}^{-1}$ denote the inverse translational operator defined on a signal $\boldsymbol{x}(t)$ as

$$\mathfrak{T}_{\tau}^{-1}\boldsymbol{x}(t) = \boldsymbol{x}(t-\tau) \tag{2.9}$$

and similarly for a sub-segment of $\boldsymbol{x}$ as

$$\mathfrak{T}_{\tau}^{-1}\boldsymbol{x}_{a:b} = \boldsymbol{x}_{(a-\tau):(b-\tau)} \tag{2.10}$$

where $\tau \in \mathbb{Z}$ is the amount of shift in time (see Section 1.6).

For each item in the original dataset, we augment the training dataset with translated copies of that item covering the whole support[14] of the time index t

$$\tilde{\mathfrak{D}} = \bigcup_{(\boldsymbol{x}_{t-d:t},\, y)\in\mathfrak{D}} \left\{ \left(\mathfrak{T}_{\tau}^{-1}\boldsymbol{x}_{t-d:t},\, y\right) \;\middle|\; \tau = 0,\, 1,\, 2,\, \cdots,\, d-1 \right\} \tag{2.11}$$

[12]In general, any operation controlling the phase t so as to render it independent of the load label would suffice, and phase alignment (that is, aligning all measurements to a fixed phase) is just an example.

[13]Which has to hold in test time though, and requires deployment-time phase alignment to satisfy the common assumption of identical distributions from training to test.

[14]Assuming a periodic signal such that $\boldsymbol{x}(t) = \boldsymbol{x}(t+nT)$ for some positive integer n, we refer to the set $0, 1, \dots, T-1$ as the support of the phase shift since no other value for the phase t can produce further variations in the waveform.

In words, $\tilde{\mathscr{D}}$ comprises all possible d-length sub-segments from the last two periods of each measurement (see Figure 2.4). Observe that the load label y remains invariant to the translation index τ which affects only the phase shift of the waveform. This is the key ingredient we intend for the data-driven model to learn.

Equation (2.11) shows an extreme case of translation invariance data augmentation in which training data is expanded by unit-step translations given by $\tau = 0, 1, 2, \dots, d-1$ and the original dataset is expanded by a factor of d. In this case, all possible values of t are observed exactly once for each data sample rendering the conditional probability $p(t \mid y)$ a discrete uniform distribution over the support $\{0, 1, \dots, d-1\}$ (i.e. independent of y) such that $p(t \mid y) = 1/d$ for all y. Stated equivalently, the marginal probability $p(t)$ becomes

$$p(t) = \sum_{y \in \mathbb{Y}} p(t \mid y)\, p(y) = \frac{1}{d} \sum_{y \in \mathbb{Y}} p(y) = \frac{1}{d} = p(t \mid y)$$

for any y, and that satisfies the targeted independence $t \perp\!\!\!\perp y$.

For practical feasibility, we rather consider expanding the dataset with ϵ-step translations such that $\tau = 0, \epsilon, 2\epsilon, \dots$ which expands the dataset by a factor of d/ϵ (rather than d). This introduces the second hyperparameter ϵ whose objective is to balance a trade-off between computational feasibility and translational invariance. This augmentation parameter is set to $\epsilon = 10$ steps in all our experiments. We denote this ϵ-dependent relaxed expansion by $\tilde{\mathscr{D}}_\epsilon$ defined as

$$\tilde{\mathscr{D}}_\epsilon = \bigcup_{(\boldsymbol{x}_{t-d:t},\, y) \in \mathscr{D}} \left\{ \left(\mathfrak{T}_\tau^{-1} \boldsymbol{x}_{t-d:t},\, y \right) \,\middle|\, \tau = 0, \epsilon, 2\epsilon, \dots \text{ and } \tau \leqslant d-1 \right\} \tag{2.12}$$

where $\epsilon \in \{1, 2, \dots, d\}$ is a positive integer with $\epsilon = 1$ denoting perfect augmentation with the highest computational cost while $\epsilon = d$ denotes no augmentation. We discuss the effects of this relaxation on the performance of the trained model in Section 2.7.7.

Figure 2.4 visually illustrates the data augmentation procedure using the two periods of the normalized current signal from a selected load in PLAID.

In a nutshell, data augmentation provides a larger set of training data diminishing the effect of overfitting, eliminates the need for phase alignment of the signals, and more importantly renders a fully connected neural network invariant to the initial

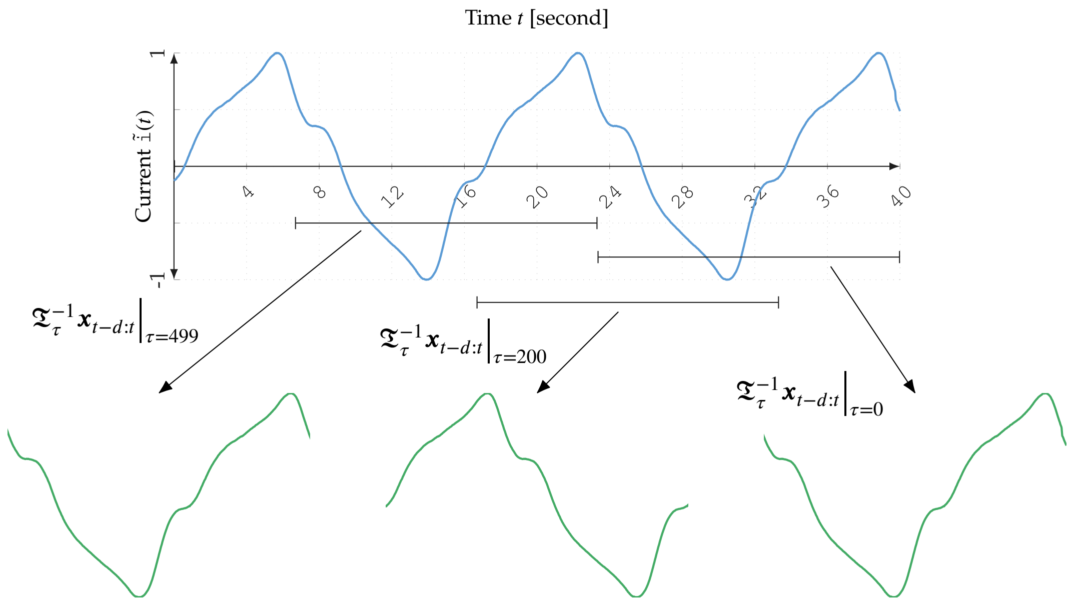


Figure 2.4.: Illustration of the adopted artificial expansion of training data from two periods of an end-use load (air conditioner) in order to enforce translation-invariant prediction in the adopted models. The sampling rate is 30 kHz, grid frequency is 60 Hz, and dimensionality is, therefore, $d = 500$.

phase of the extracted signatures. As a result, the model becomes more *translation-invariant* and, accordingly, more robust to signal variations during steady-state operation of appliances.

As noted earlier [Baptista et al. 2018, Davies et al. 2019], such translation invariance is also achievable using a transitional classifier (e.g. dense stack of neural net layers) fitted on top of a stack of translation-equivariant (e.g. convolutional) layers LeCun et al. [1998a], Krizhevsky et al. [2012].

2.5. Model Architecture

In this section, we introduce our proposed model for real-time load identification based on an ensemble of small neural networks. First, we introduce the architecture of a single base model in Section 2.5.1. Afterwards, we detail in Section 2.5.2

how base models are integrated in an ensemble for a final decision on the predicted load category. We then discuss relevant remarks and advantages of the proposed architecture in Section 2.5.3.

For notational convenience and without loss of generality, we formulate the proposed model based on the original dataset $\mathscr{D}$ as defined in Equation (2.6), but the formulation remains valid for any augmented set $\tilde{\mathscr{D}}_\epsilon$.

2.5.1. The Base Model

The proposed model is an ensemble of small binary neural nets each referred to as a *base model* and a final committee decision mechanism. Prior to detailing the complete model architecture, we first describe the functionality and architecture of the base model.

Loosely stated, the base model is a small binary neural network targeted with distinguishing between particularly two different load categories. The architecture remains identical across all base models, but each pair of class labels leads to a particular set of parameter values[15]. Within a given pair of class labels, the output of the corresponding base model is interpreted as the likelihood of a particular one of the two labels knowing that the true class is either of the two.

Formally, let y_i and y_j be a selected pair of class labels such that $y_i, y_j \in \mathbb{Y}$. Let further $g^{(i,j)} : \mathbb{R}^{2d\times 1} \to [0, 1]$ denote a mapping from a load signature $\boldsymbol{x}_{t-d:t} \in \mathbb{R}^{2d\times 1}$ (that is, the $2d$-dimensional VI-waveform of a load) to the parameter $p = g^{(i,j)}(\boldsymbol{x}_{t-d:t})$ of a Bernoulli distribution so that

$$p\big(y_i \;\big|\; \boldsymbol{x}_{t-d:t},\, y \in \{y_i, y_j\}\big) = \mathrm{Bernoulli}(p) = \mathrm{Bernoulli}\big(g^{(i,j)}(\boldsymbol{x}_{t-d:t})\big) \tag{2.13}$$

where $p(A \mid B)$ denotes the conditional probability of the event (or random variable) A on the event B assuming it has occurred. Equation (2.13) is interpreted as how likely it is for the true load category to be y_i given its load signature $\boldsymbol{x}_{t-d:t}$ and the knowledge that it can only be either of the two classes y_i or y_j based on the adopted selection mechanism.

[15]Parameters of a neural net are generally weights and biases of each linear (or affine) transformation.

During training, the selection mechanism that guarantees the assumed condition $y \in \{y_i,\ y_j\}$ is simply training that specific base model on the subset $\mathscr{B}^{(i,j)}$ comprised exclusively of y_i and y_j load categories

$$\mathscr{B}^{(i,j)} = \left\{ (\boldsymbol{x}_{t-d:t},\ y) \in \mathscr{D} \;\middle|\; y \in \{y_i, y_j\} \right\} \tag{2.14}$$

Upon deployment though, no selection is supposed to take place, and hence the second condition $y \in \{y_i,\ y_j\}$ is in some cases violated. In other words, a given test sample $\boldsymbol{x}^*_{t-d:t}$ may belong to any of the M target classes while the model has been trained on the two ith and jth categories. It is unknown a priori whether the given test sample $\boldsymbol{x}^*_{t-d:t}$ indeed belongs to the training data distribution for that specific base model or it is an out-of-distribution sample. In fact and as Section 2.5.2 will illustrate, a test sample $\boldsymbol{x}^*_{t-d:t}$ whose real load label is e.g. y^* is an out-of-distribution sample for exactly $\binom{M}{2} - (M-1)$ base models which never observed the class y^* in their selected training sets $\mathscr{B}^{(i,j)}$.

Nevertheless, the hope is that the trained model exhibits an abstaining behavior (such that $g^{(i,j)}(\boldsymbol{x}^*_{t-d:t}) \to 0.5$) for out-of-distribution samples stating that it is *uncertain* about its own prediction based on the the current sample since it belongs to a class outside its two-element label space $\{y_i,\ y_j\}$. In other words, the model is supposed to be highly certain of its prediction

$$\left| g^{(i,j)}(\boldsymbol{x}_{t-d:t})^* - 0.5 \right| \to 0.5 \quad \text{if } y^* \in \{y_i,\ y_j\} \tag{2.15}$$

if the condition holds, and uncertain otherwise

$$\left| g^{(i,j)}(\boldsymbol{x}_{t-d:t})^* - 0.5 \right| \to 0 \quad \text{if } y^* \notin \{y_i,\ y_j\} \tag{2.16}$$

assuming the model's sigmoidal activation is an indicator of its predictive uncertainty[16]. Such a behavior is better observed in Bayesian models (e.g. Bayesian neural nets (BNN)[17]) [Gal 2016] but we delegate this approach to future work.

As mentioned earlier, a base model $g^{(i,j)}$ is a shallow neural network whose architecture is defined by two affine transformations each followed by a sigmoidal acti-

[16]Gal [2016], for example, illustrates that a sigmoidal function is not a valid indicator of uncertainty.
[17]That is, neural networks with stochastic synaptic connections (aka weights or parameters).

vation function

$$g^{(i,j)}(\boldsymbol{x}_{t-d:t}) = \sigma\Big(\mathbf{w}_2^{(i,j)} \tanh\left(\mathbf{W}_1^{(i,j)}\boldsymbol{x}_{t-d:t} + \boldsymbol{b}_1^{(i,j)}\right) + b_2^{(i,j)}\Big) \quad \in [0,1] \tag{2.17}$$

where $\sigma(x) = (1+\exp(-x))^{-1}$ is the logistic function and $\tanh(x) = 2\sigma(2x)$ is the hyperbolic tangent (with their multivariate variants defined element-wise). In other words, we consider a 3-layer neural network with d-neuron input layer, a single 30-neuron hidden layer, and a single-neuron output layer. Model parameters comprise the weights $\mathbf{W}_1^{(i,j)} \in \mathbb{R}^{30\times 2d}$, $\mathbf{w}_2^{(i,j)} \in \mathbb{R}^{1\times 30}$ and the biases $\boldsymbol{b}_1^{(i,j)} \in \mathbb{R}^{30\times 1}$, $b_2^{(i,j)} \in \mathbb{R}$ of each affine transformation, leading to a model size (quantified as number of trainable parameters) of $60d + 61$ parameters. For PLAID with $F_s = 30$ kHz, $F_g = 60$ Hz, and $d = F_s/F_g = 500$ samples, the resulting base model size is $\sim 30\times 10^3$ parameters. The model size is a linear function of (and mostly affected by) the sampling frequency F_s.

With the selected subset $\mathscr{B}^{(i,j)}$ and the binary cross-entropy loss function[18], (details on which are delegated to Chapter 5 for a more elaborate discussion on a similar approach) a base model is trained so as to maximize the log-likelihood of observing the given training dataset. We utilized `matlab`'s implementation of the conjugate gradient descent with random restarts [Powell 1977] as an optimizer for all models. A final remark is that, we only train a base model $g^{(i,j)}$ for each $i < j$ pair since we can utilize its reflection $1 - g^{(i,j)}$ as the likelihood of the second class label y_j (that is $p\big(y_j \;\big|\; \boldsymbol{x}_{t-d:t},\, y \in \{y_i, y_j\}\big) = \text{Bernoulli}\big(1 - g^{(i,j)}(\boldsymbol{x}_{t-d:t})\big)$) under identical conditions.

In Figure 2.5, a base model consists of the input layer (left-most) and a single row of the middle layers (a hidden layer and an output layer). Next, we discuss how the whole ensemble is constructed.

2.5.2. The Ensemble Architecture

The ensemble architecture comprises a one-vs-one realization of the multi-class classification task using binary classifiers. A base model $g^{(i,j)}$ is trained on an appropri-

[18]Later work (e.g. [Gast and Roth 2019]) showed empirically that using a linear layer at the output (instead of a logistic sigmoid) and sum of squared error (SSE) as a loss function indeed drives the network towards the targeted range, while mitigating the vanishing gradient problem. In our shallow networks, however, the problem of vanishing gradients was minimally, if ever, observed.

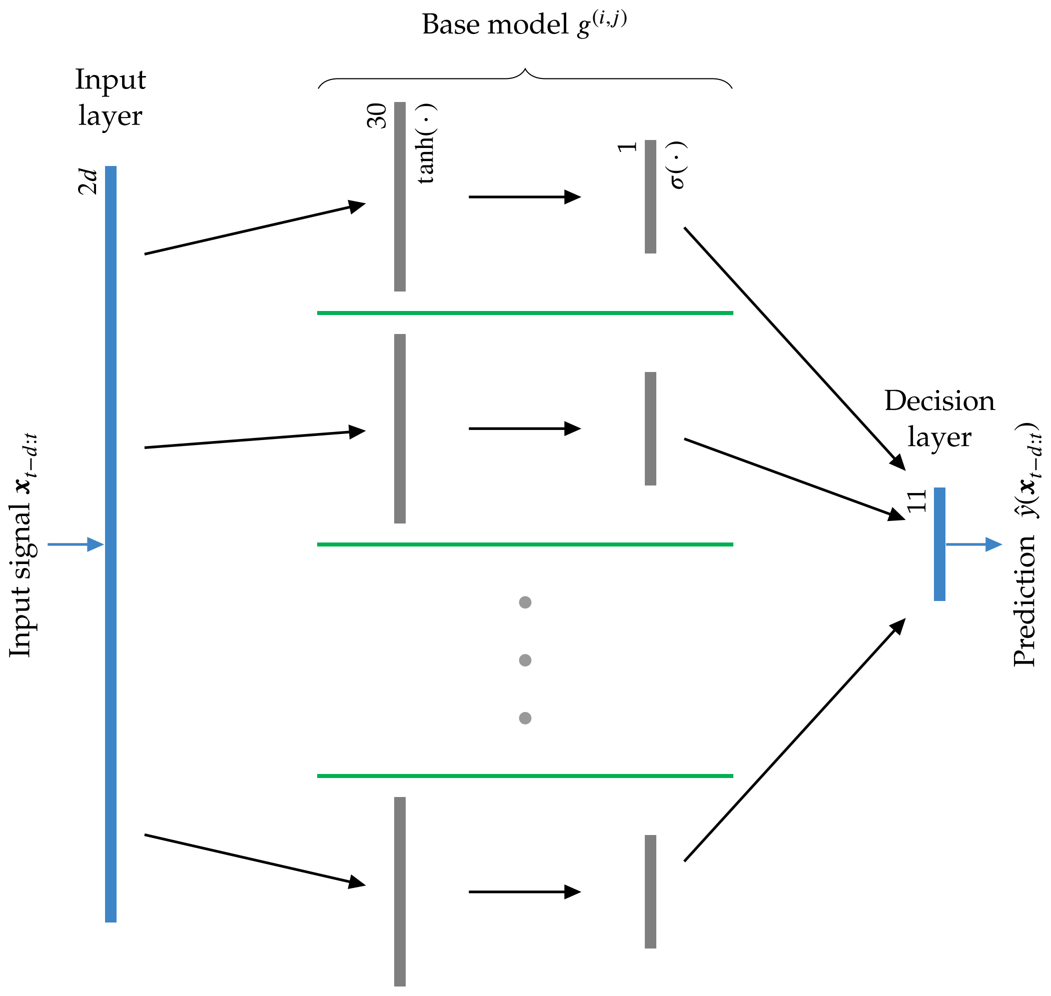


Figure 2.5.: Illustration of the proposed model architecture. The input signal $\boldsymbol{x}_{t-d:t}$ is a VI-waveform signature of a load. This input is then distributed to an ensemble of 1-vs-1 base models. Base model outputs are then aggregated for a final prediction in the output decision layer (e.g. majority voting or maximum confidence). Training (i.e. parameter optimization) is performed only on the base model level. A single base model consists of the input layer (left-most) along with a single row of the middle layers (see Equation (2.17)). Layer sizes (in dimensionality) is noted top left of each layer, and its non-linear activation (if exists) is noted top right. The activations $\sigma(\cdot)$ and $\tanh(\cdot)$ denote the logistic and hyperbolic tangent sigmoidal functions, respectively.

ate subset of the training data (that is, $\mathscr{B}^{(i,j)}$) which comprises exclusively samples of either of the two class labels y_i and y_j for each class combination y_i and y_j where $i < j$. Accordingly, a total of $\binom{M}{2}$ base models are trained for M load categories using the aforementioned training procedure where $\binom{\cdot}{\cdot}$ is the binomial coefficient.

From the ensemble of $\binom{M}{2}$ base models, we reach a final class prediction using the committee decision layer. An example of a committee decision function includes *majority voting*. Majority voting counts the number of class votes across base models for each target class, where a class vote for class y_i against y_j given the load signature $\boldsymbol{x}_{t-d:t}$ is defined as $\mathbb{1}(g^{(i,j)}(\boldsymbol{x}_{t-d:t}) \geq 0.5)$ if $i < j$ and $\mathbb{1}(g^{(i,j)}(\boldsymbol{x}_{t-d:t}) \leq 0.5)$ if $i > j$ where $\mathbb{1}(\,\cdot\,)$ is the indicator function. The total class vote for the i^{th} class across all base models is given by

$$g^{(i)}(\boldsymbol{x}_{t-d:t}) = \sum_{j \neq i} \mathbb{1}\left(g^{(i,j)}(\boldsymbol{x}_{t-d:t}) \geq 0.5\,\mathrm{sgn}(j-i)\right) \tag{2.18}$$

where $\mathrm{sgn}(\,\cdot\,)$ is the sign function. The final class prediction can be then defined as

$$\hat{y}(\boldsymbol{x}_{t-d:t}) = \underset{y_i \in \mathbb{Y}}{\arg\max}\; g^{(i)}(\boldsymbol{x}_{t-d:t}) \tag{2.19}$$

which means that the estimated class is the one with the most votes.

Another example of a final committee decision mechanism that leverages the probabilistic interpretation of base model scores, is the *maximum confidence* voting in which the final class score is an average of the base model predictions $g^{(i,j)}$ interpreted as their confidence scores

$$g^{(i)}(\boldsymbol{x}_{t-d:t}) = \frac{1}{M-1}\left(\sum_{i<j} g^{(i,j)}(\boldsymbol{x}_{t-d:t}) + \sum_{i>j}\left(1 - g^{(i,j)}(\boldsymbol{x}_{t-d:t})\right)\right) \tag{2.20}$$

and the final class prediction remains the same as in Equation (2.19).

This adopted one-vs-one scheme indeed represents one of the scalability limitations of such an architecture. In other words, one-vs-one multi-class generalization of binary classifiers scales quadratically with the number of classes. This is treated in this work simply by adopting small-sized base models.

As an example, PLAID contains 11 load categories leading to a total of 55 base models, each comprising its own parameters resulting in a total model size of $0.5(M^2 -$

$M)(60d + 61) \simeq 1.65 \times 10^6$ trainable parameters for the sampling frequency $F_s = 30$ kHz. Nevertheless, the total training time of all 55 base models is $1 \sim 2$ hours on a modern 12GB-GPU unit. Model size and training time are notably reduced for lower sampling rates or reduced number of classes.

Finally, we utilize a validation-based early stopping as a regularization technique to avoid over-fitting [Prechelt 2012]. As mentioned earlier, PLAID measurements contain, for each load category, several instances from various households. In order to improve model generalization to new buildings, a predefined fraction of households assigned originally for training is reserved for validation[19]. Training and validation subsets are mutually exclusive building-wise. Please note that, this is only a discussion of how the training set is split for gradient-based optimization and early stopping. For evaluating our model, a suitable cross validation scheme is adopted on a prior level as shall be discussed in the rest of this chapter.

Figure 2.5 illustrates the architecture of the proposed model. Leftmost is the input layer representing the queried load signature $\boldsymbol{x}_{t-d:t}$ which is then distributed to all base models $g^{(i,j)}$ whose outputs are aggregated in a rightmost decision layer.

2.5.3. Remarks

The proposed model is characterized by three main features, namely, modularity of the one-vs-one ensemble, *data-driven-ness* of base models, and a small-sized base model architecture. These aspects led to various advantages of the proposed model, of which we highlight the following.

First, it allows for a straightforward and simple incorporation of domain knowledge. An example of such a feature is illustrated in Section 2.7 where knowledge of the categories of loads in each target building leads to a notable boost in identification performance and reduction in total model size by simply ruling out irrelevant base model (that is, base model models whose labels includes non-existent loads). Similarly, preference can be directed to certain target loads by simple adjustments to the final decision layer (e.g. using a weighted decision function).

[19]In other words, this fraction is not used for gradient-based optimization.

Second, the adopted shallow and small architectures are more suitable for embedded system deployments, where a single sub-network is around 30×10^3 parameters in size. Training such small models with standard gradient-based optimization, even with the extensively augmented dataset, is less than a couple of minutes on a modern GPU.

Third, it is known that neural networks are one of the unstable learning algorithms with respect to training data [Aggarwal 2014] (i.e. they are sensitive to changes in the training set). For unstable models, an ensemble approach known as Bootstrap aggregation (aka Bagging) [Breiman 1996, Friedman and Hall 2007] is expected to provide more robust results than a single base model[20]. Our adopted ensemble framework shares numerous similarities with Bagging.

We also highlight some limitations of the adopted architecture that we propose for future work on the topic in Section 2.8.

2.6. Performance Assessment Measures

In this section, we introduce the adopted measures for empirically quantifying the performance of our model on a benchmarking dataset. The adopted metrics represent one generalization of the standard binary classification measures [Powers 2011]. These standard measures are detailed in Section 5.5 and the unfamiliar reader will benefit from consulting that section prior to the present one.

Let $\mathbf{\Lambda} = [\Lambda_{ij}]_{i,j=1}^{M,M}$ denote the $M \times M$ square confusion matrix whose i^{th}-row j^{th}-column element Λ_{ij} represents the number of data samples whose true class label is $y = y_i$ and model prediction $\hat{y} = y_j$ over some evaluation dataset $\mathscr{D}^{(\text{test})}$ with $N^{(\text{test})}$ data samples, denoted hereafter simply by $\mathscr{D}$ with N for brevity[21]. Let further N_m denote the number of occurrences of class y_m in the target population.

[20]In Bootstrap aggregation, an ensemble of weak, multi-class learners, each trained on a randomly sampled subset of the whole training dataset, is used rather than binary-classifiers generalized to a multi-classification problem.

[21]This should not cause any confusion with the training set denoted earlier by $\mathscr{D}$ since we primarily consider the evaluation set hereafter. Should a confusion arise though, we will revert back to distinctly superscripted denotations $\mathscr{D}^{(\text{train})}$ and $\mathscr{D}^{(\text{test})}$ for the training and evaluation sets, respectively.

We define the per-class performance indicators as

$$\begin{aligned}
\text{True Positives: } \texttt{TP}_m &= \sum_{(\boldsymbol{x}_{t-d:t},\, y)\in\mathscr{D}} \mathbb{1}\big(\, \hat{y}(\boldsymbol{x}_{t-d:t}) = y_m \;\&\; y = y_m\big)\\
\text{True Negatives: } \texttt{TN}_m &= \sum_{(\boldsymbol{x}_{t-d:t},\, y)\in\mathscr{D}} \mathbb{1}\big(\, \hat{y}(\boldsymbol{x}_{t-d:t}) \neq y_m \;\&\; y \neq y_m\big)\\
\text{False Positives: } \texttt{FP}_m &= \sum_{(\boldsymbol{x}_{t-d:t},\, y)\in\mathscr{D}} \mathbb{1}\big(\, \hat{y}(\boldsymbol{x}_{t-d:t}) = y_m \;\&\; y \neq y_m\big)\\
\text{False Negatives: } \texttt{FN}_m &= \sum_{(\boldsymbol{x}_{t-d:t},\, y)\in\mathscr{D}} \mathbb{1}\big(\, \hat{y}(\boldsymbol{x}_{t-d:t}) \neq y_m \;\&\; y = y_m\big)
\end{aligned}$$

for each class $y_m \in \mathbb{Y}$ where $\hat{y}(\boldsymbol{x}_{t-d:t})$ is the class prediction for the sample $\boldsymbol{x}_{t-d:t}$. These measures reduce the confusion matrix $\boldsymbol{\Lambda}$ to a standard 2×2 aggregated score matrix for each class y_m from which normalized measures can be estimated. Normalized measures for the m^{th} class are then defined as

$$\text{recall}_m = \texttt{TPR}_m = \frac{\texttt{TP}_m}{\texttt{TP}_m + \texttt{FN}_m} \tag{2.21}$$

$$\text{precision}_m = \texttt{PPV}_m = \frac{\texttt{TP}_m}{\texttt{TP}_m + \texttt{FP}_m} \tag{2.22}$$

$$\text{specificity}_m = \texttt{TNR}_m = \frac{\texttt{TN}_m}{\texttt{TN}_m + \texttt{FP}_m} \tag{2.23}$$

$$\text{F}_1^m\text{-score} = \texttt{F1S}_m = \frac{2\,\texttt{TP}_m}{2\,\texttt{TP}_m + \texttt{FP}_m + \texttt{FN}_m} \tag{2.24}$$

where TPR is the $\underline{\text{t}}$rue $\underline{\text{p}}$ositive $\underline{\text{r}}$ate or recall, PPV $\underline{\text{p}}$ositive $\underline{\text{p}}$redictive $\underline{\text{v}}$alue or precision, and TNR is the $\underline{\text{t}}$rue $\underline{\text{n}}$egative $\underline{\text{r}}$ate or specificity.

The per-class metrics are further aggregated to a single scalar-valued measure via unweighted average leading to the macro-F_1-score defined as

$$\text{macro-F}_1\text{-score} = \frac{1}{M}\sum_{m=1}^{M} \text{F}_1^m\text{-score} \tag{2.25}$$

or the class weighted average known as weighted-F_1-score and given by

$$\text{weighted-F}_1\text{-score} = \sum_{m=1}^{M} \frac{N_m}{N}\, \text{F}_1^m\text{-score} \tag{2.26}$$

Additionally, we utilize the scalar-valued unweighted accuracy metric α

$$\alpha = \frac{\sum_i \Lambda_{i,i}}{\sum_{i,j} \Lambda_{i,j}} = \frac{\operatorname{tr}(\boldsymbol{\Lambda})}{\mathbf{1}_{1,M}\boldsymbol{\Lambda}\mathbf{1}_{M,1}} = \frac{\operatorname{tr}(\boldsymbol{\Lambda})}{N} \tag{2.27}$$

and Cohen's Kappa κ measure [Ben-David 2008] given by

$$\kappa = \frac{\alpha - p_e}{1 - p_e} \tag{2.28}$$

for an overall performance assessment and benchmarking against prior art and related work, where p_e is the expected agreement with the prior class distribution, $\mathbf{1}_{N,M}$ is the $N \times M$ matrix of all ones, and $\operatorname{tr}(\cdot)$ is the matrix trace operator.

Loosely speaking, the main distinction between the accuracy metric α and Cohen's kappa κ is the metric's baseline. The former is normally benchmarked against a strictly random classifier that simply assigns absolutely random guesses in each prediction with no prior information (i.e. assuming equiprobable classes) $p(y \,|\, \boldsymbol{x}) = 1/M \;\forall y \in \mathbb{Y}$. The latter, however, quantifies the improvement against a naïve classifier that leverages prior class distributions in its random guesses $p(y \,|\, \boldsymbol{x}) = p(y)$. Cohen's Kappa κ was, therefore, originally proposed in order to lessen the biasedness of the accuracy, precision, and recall measures in highly imbalanced class distributions [Cohen 1960, Aggarwal 2014].

The expected agreement is defined as

$$p_e = \underbrace{\frac{\mathbf{1}_{1,M}\boldsymbol{\Lambda}}{N}}_{\text{classifier marginals } p(\hat{y})} \cdot \underbrace{\frac{\boldsymbol{\Lambda}\mathbf{1}_{M,1}}{N}}_{\text{empirical priors } p(y)} = \frac{\operatorname{tr}(\boldsymbol{\Lambda}\mathbf{1}_{M,M}\boldsymbol{\Lambda})}{N^2} \tag{2.29}$$

whose first term is the estimated class marginals $p(\hat{y})$ of the classifier after marginalizing out all load signatures (which is, empirically, the normalized row sum of the confusion matrix, or intuitively how often the classifier predicts a certain class regardless of the load signature). The second term (a column sum of the confusion matrix) represents how often this class indeed appears in a real dataset (that is, the empirical priors). Accordingly, the kappa measure κ is defined as

$$\kappa = \frac{N \cdot \operatorname{tr}(\boldsymbol{\Lambda}) - \operatorname{tr}(\boldsymbol{\Lambda}\mathbf{1}_{M,M}\boldsymbol{\Lambda})}{N^2 - \operatorname{tr}(\boldsymbol{\Lambda}\mathbf{1}_{M,M}\boldsymbol{\Lambda})} \tag{2.30}$$

2.7. Experiments and Results

In this section, we evaluate and empirically investigate the performance of our proposed model on a real-world energy dataset. First, the dataset is introduced in Section 2.7.1. Specifications of the proposed model for this specific dataset are detailed in Section 2.7.2. Afterwards, we detail the adopted evaluation framework in Section 2.7.3 and report our empirical results. We then discuss further investigations of the model's robustness against training data scarcity in Section 2.7.5, reduced sampling frequency in Section 2.7.6, and phase shift of load signatures in Section 2.7.4. Section 2.7.8 discusses reported results and compares our contribution to prior art on the addressed problem.

2.7.1. The PLAID Dataset

We evaluate our proposed model on the publicly available dataset named plug-level appliance identification dataset (PLAID) [Gao et al. 2014]. PLAID comprises $\sim 10^3$ instantaneous voltage and current measurements of $M = 11$ household load categories. Each load category is represented by a variety of load instances with different models across 55 residential buildings located in the US. Table 2.1 lists all load categories in PLAID along with the load instance count per category and the ratio of each category in the dataset.

Acquired measurements are on the plug-level of each load (i.e. single load operation), and sampled at $F_s = 30$ kHz where the standard grid-frequency for domestic end-use in the US is $F_g = 60$ Hz. Each measurement in PLAID comprises a 1- to 13-second window of voltage and current during (or shortly after) the turn-on transient of a load. Since transient operation is outside the scope of this work, training and evaluation sets are extracted from merely the last two periods of each measurement[22].

[22]An exception though, is the final evaluation setups (see Section 2.7.6) which extracts the evaluation dataset from the last second of each measurement.

Table 2.1.: List of load categories, instance counts, and ratio of instances in each category in the PLAID dataset [Gao et al. 2014].

Load category	Instance count N_m	Weight $\frac{N_m}{N}$
Air conditioner (AC)	66	0.062
Compact fluorescent lamp (CFL)	175	0.163
Fan	115	0.107
Fridge (FR)	38	0.035
Hairdryer	156	0.145
Heater	35	0.033
Bulb	114	0.106
Laptop	172	0.160
Microwave (MC)	139	0.130
Vacuum cleaner (VC)	38	0.035
Washing machine (WM)	26	0.024

2.7.2. Model Specifications

Since PLAID comprises $M = 11$ load categories, there exist $\binom{M}{2} = 55$ class combinations y_i and y_j where $i < j$ and for each, we train a distinct base model. A base model as formulated by Equation (2.17) and depicted in Figure 2.5 comprises a two-layer, fully connected, feed-forward neural network with $2d$ input neurons, 30 hidden neurons, a single output neuron, and sigmoidal non-linear activations.

For translation invariance, the extracted training set is augmented according to Equation (2.12) with a translation step of $\epsilon = 10$. Consequently, the training dataset is expanded by an expansion factor of $d/\epsilon = 50$ for the dimensionality $d = 500$. For lower dimensionalities (e.g. resulting from downsampling as in Section 2.7.6) the translation step ϵ is adjusted so as to keep the expansion factor d around the same value for all experiments.

2.7.3. Evaluation Framework

For a fair comparison of our model with prior works that were empirically evaluated on PLAID, we adopt the same cross validation scheme as in [Gao et al. 2015], namely leave-one-out cross validation where "one" here denotes a single building. In other words, measurements from a certain building (the test building) are held out for evaluation in an evaluation dataset $\mathscr{D}^{(\text{test})}$ while the remaining set of measurements (across all other 54 buildings) are available to the model during the training phase in the training set[23] $\mathscr{D}^{(\text{train})}$. The process is repeated for each building in PLAID resulting in a total of 55 evaluation folds. This cross validation scheme follows a practical use of trained models, because the long-sought goal in all these models is, in fact, to generalize deployments to new previously unseen buildings.

With the leave-one-out cross validation on PLAID buildings, we report our main results in Figure 2.6 where per-class performance is shown, and Figure 2.7 where aggregate measures are reported on a per-household basis. Additionally, Table 2.2 reports our best achieved aggregate measures of accuracy α, macro–F_1-score, and weighted-F_1-score. The best result obtained for the maximum confidence decision mechanism (see Equation (2.20)) is $\alpha = 89.7\%$ while the unweighted majority voting (see Equation (2.18)) achieved 88.2% of accuracy. We report the final accuracy of this study as 88.92%(±0.745) for an argument elaborated in Section 2.7.7.

It is observed from the figures that microwaves, compact fluorescent lamps, vacuum cleaners, and laptops are the most identifiable loads whereas temperature control devices (such as air conditioners, heaters, fridges, and even fans) are the least. Upon relating these two sets to the load signatures depicted in Figure 2.3, one observed a relationship between a load's prediction score and non-linearities exhibited in its signature. This supports our claim earlier that non-linearities in VI-waveforms is expected to promote identifiability but are tackled by reliable feature engineering and extraction.

2.7.4. Leveraging External Knowledge

As discussed earlier, one key advantage of the adopted ensembling architecture (i.e. the one-vs-one construction) is ease of incorporation of domain knowledge. For an illustrative example, we emulate this process through partial knowledge of targeted load categories in a building which maps to a reduction in the label space[24] $\mathbb{Y}$ for that specific building and correspondingly a reduction in the model size. In other words, if a specific building is known to be empty of air conditioners, the reduced

[23]The training $\mathscr{D}^{(\text{train})}$ is further partitioned into a validation set for early stopping and a sub-training set for gradient estimations.

[24]The reduced label space $\tilde{\mathbb{Y}}$ is building-specific, but we omit this dependence in notation for convenience.

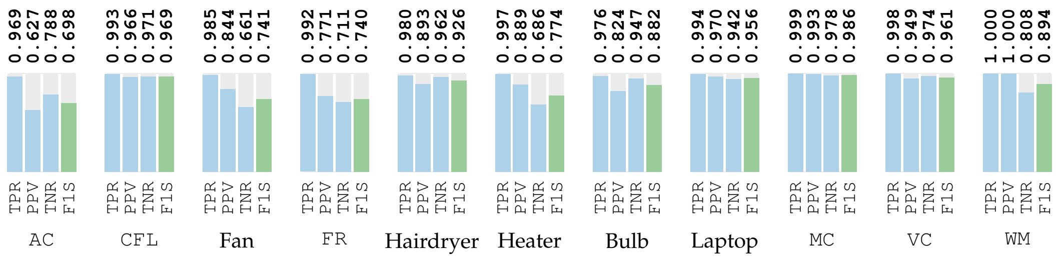


Figure 2.6.: Per-load measures $\mathtt{TNR}_m$, $\mathtt{PPV}_m$, $\mathtt{TPR}_m$, $\mathtt{F1S}_m$ for F_s = 30 kHz.

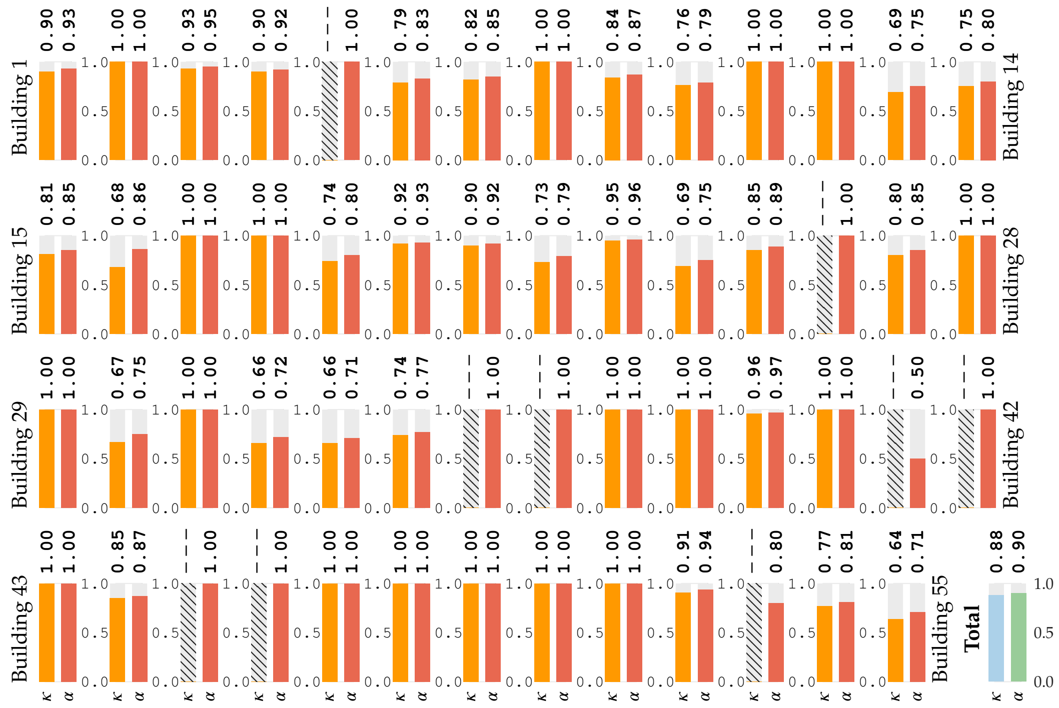


Figure 2.7.: Per-building evaluation measures for all loads. Note that Cohen's Kappa can return an undefined value (for a zero-valued denominator) and this is observed in the cases of buildings containing a single load category. This applies to the 26th, 46th, and 53rd buildings from which only washing machine, microwave, and laptop measurements were collected, respectively. The 5th, 35th, 41st, 42nd, and 45th are all limited to CFL measurements.

label space for this building becomes $\tilde{\mathbb{Y}} = \mathbb{Y} - \{\text{'AC'}\}$ and we rather train 45 binary networks. While this process is valid for any model (e.g. CNN), the advantage in our construction is that reduction in the total model size is quadratic with respect to label space reductions, and is (for the one-vs-one construction) given by $(M-1)\Delta - \Delta^2$ where Δ is the reduction in label space (i.e. the number of load categories known to not exist in the target building). Leveraging external knowledge, of course, improves the performance where the best-case accuracy reaches $\alpha = 94\%$.

For all upcoming experiments, we repeat each experiment twice. The first is performed using the whole label space $\mathbb{Y}$ and the second exploits external knowledge in reducing the label space $\tilde{\mathbb{Y}}$ using the list of appliances in each household. This is depicted in the right and left sub-figures, respectively, of Figures 2.8 to 2.10.

2.7.5. Training under Data Scarcity

In the first study, we investigate the effect of training data size on the performance of the proposed model and how it performs under extreme situations of data scarcity. To this end, we repeatedly reduce the size of the training dataset and estimate the total performance using the same cross validation technique for each reduction.

In each test case, a subset $\mathscr{B}(r) \subseteq \mathscr{D}^{(\text{train})}$ is randomly sampled from the training set $\mathscr{D}^{(\text{train})}$ such that $r = |\mathscr{B}| / |\mathscr{D}^{(\text{train})}|$ represents the ratio of the subset size to the whole training set where $|\mathscr{A}|$ denotes the cardinality of the set $\mathscr{A}$. The training subset $\mathscr{B}(r)$ is sampled building-wise such that either all samples from a building are included in $\mathscr{B}(r)$ or none, until the targeted size ratio r is reached. This approach emulates a more realistic data scarcity condition (where certain households reject to participate in data acquisition) than randomly discarding individual measurements. Varying the subset size r permits emulating data scarcity situations on which we evaluate the proposed model.

Figure 2.8 shows the effect of training data reduction (starting from extreme cases of only 100 data samples $r \simeq 0.1$ to the complete training set $r = 1.0$). As expected, the performance of the model degrades notably as the size of the training data decreases but notable also is the approximate linearity of this degradation through a wide range $r \in [0.2, 0.9]$ of training data reductions.

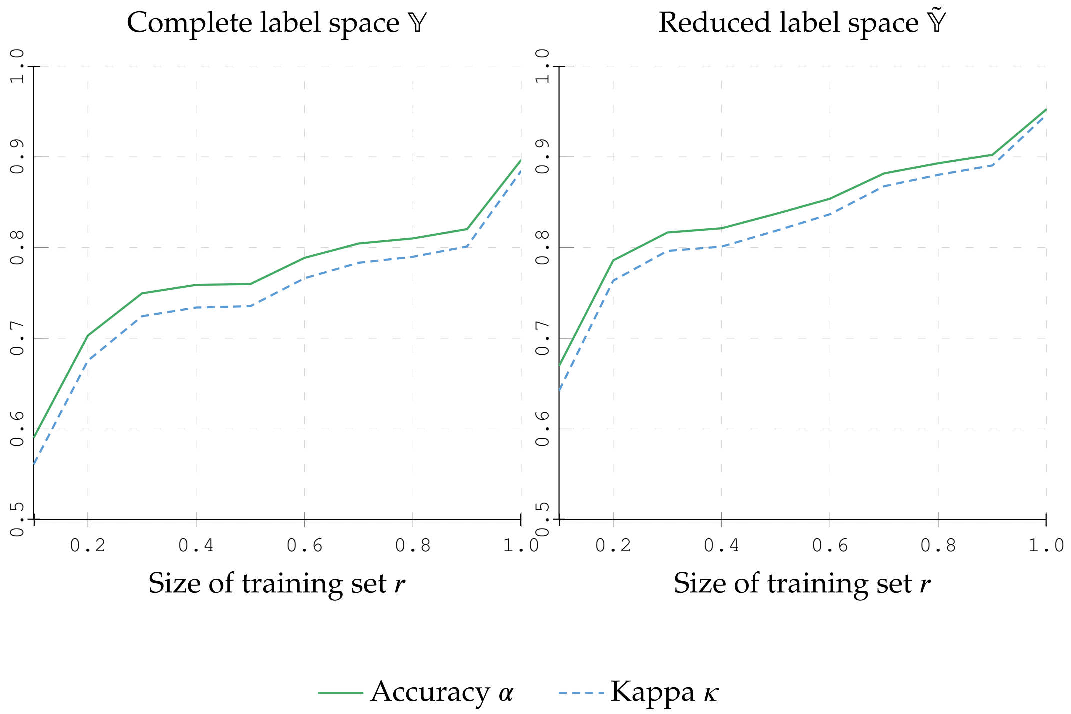


Figure 2.8.: An empirical study (using the aggregate evaluation measures: accuracy α and kappa κ) of the sensitivity of the trained models to the size of the training dataset. In the figures, r is the ratio of the randomly sampled subset $\mathcal{B}(r) \subseteq \mathcal{D}^{(\text{train})}$ to the size of the canonical training set $\mathcal{D}^{(\text{train})}$. A reduced label space $\tilde{\mathbb{Y}}$ is where the list of load categories in the target building is known in advance and missing classes are excluded from the final committee decision mechanism.

2.7.6. Identifiability under Reduced Sampling Rates

In the second study, we use the complete training dataset (i.e. $r = 1$) but rather reduce the sampling rate of the raw measurements F_s through downsampling. Downsampling is performed in three stages for which we utilized `matlab`'s implementation of a sample rate converter given by the function `mfilt.firsrc`. First the closest rational number to the ratio $F_s/\tilde{F}_s$ is estimated such that

$$\frac{F_s}{\tilde{F}_s} \simeq \frac{a}{b} \in \mathbb{Q} \tag{2.31}$$

where $\tilde{F}_s$ is the target frequency, and $a,\ b \in \mathbb{N}$. The signal is first upsampled by a factor a, then filtered using a suitable finite impulse response (FIR) filter to avoid aliasing, and finally decimated by a factor b to reach the targeted sampling frequency $\tilde{F}_s$.

For different sampling frequencies $\tilde{F}_s$, the study reveals the robustness of our method with respect to changes in the sampling frequency. As observed in Figure 2.9, the accuracy slightly drops but is always well above 80% even at 2.5 kHz. Apart from the rapid loss in performance from the original sampling frequency of 30 kHz, performance thereafter remains more stable (that is within ±2.3%) than reported in prior work [Gao et al. 2015] in which the accuracy varied in a range of ±7% within the same range of frequencies. The rapid drop in performance between 30 kHz and 25 kHz is unknown to us, but is most likely attributed to the fractional decimation procedure that was not deeply investigated in this work[25]. Further investigation is delegated to future work.

2.7.7. Temporal Stability

In this experiment, we test the sensitivity of the proposed model to the phase shift of the extracted segments. Towards that end, we constructed the following evaluation setup. For each cross validation fold (that is, for each target building) with the evaluation set $\mathscr{D} = \{(\boldsymbol{x}_{t-d:t}^{(n)},\ y^{(n)})\}_{n=1}^{N}$ comprising the last period $\boldsymbol{x}_{t-d:t}$ of each measurement, we construct the following phase-shifted set

$$\mathscr{D}(\tau) = \left\{ \left(\mathfrak{T}_\tau^{-1}\boldsymbol{x}_{t-d:t},\ y\right) \,\middle|\, \left(\boldsymbol{x}_{t-d:t},\ y\right) \in \mathscr{D} \right\} \tag{2.32}$$

where $\tau \in \mathbb{Z}_{\geqslant 0}$ is a non-negative phase shift (here $\mathbb{Z}_{\geqslant a}$ denotes the set of integers in the interval $[a, \infty)$). In $\mathscr{D}(\tau)$ each load signature is shifted by τ while retaining the same class label y. The model is then alternatively evaluated on the phase-shifted dataset $\mathscr{D}(\tau)$. For different phase shift τ, such an evaluation setup puts the sought translation invariance to the test and investigates the temporal stability of the proposed model.

[25]For example, it was not clear to us what sort of interpolation was adopted by the `matlab`'s function `mfilt.firsrc` for upsampling.

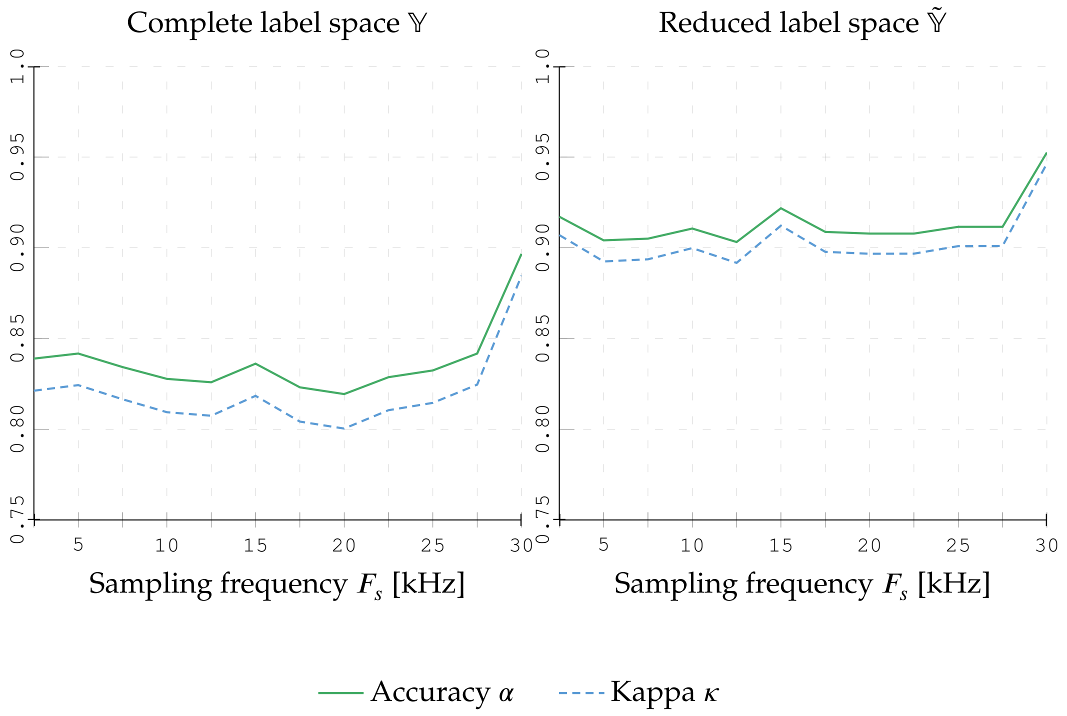


Figure 2.9.: Empirical investigation of the sensitivity of the proposed model to reductions in the sampling frequencies. Curves illustrate stable and minimal improvements upon increasing the sampling frequency except for the relatively upper range (25-30 kHz). Robustness against reduced sampling rates is clearly observed in the consistent and stable performance (81.9% to 84.2% in accuracy) for the whole range from 2.5-25 kHz.

Figure 2.10 demonstrates the behavior of the proposed model over the range of phase shifts from 0 to 1 second since PLAID comprised minimum 1-second measurements. The figure depicts two phenomena, a declining trend from 89.67% of accuracy at $\tau = 0$ to 87.05% at $\tau = 1$ seconds (that is, 30000 samples) revealing the stability of our model over a wide range of phase shifts. Worth noting here is that, loads in PLAID are not necessarily switched-*on* since the epoch of the measurement (that is, at time $t = 0$) and hence moving backwards in time increases the likelihood of querying the model with void signatures. This partially attributes the declining trend to the evaluation setup and renders the obtained results (in spite of out-performing state-of-art [Gao et al. 2015]) pessimistic to an extent.

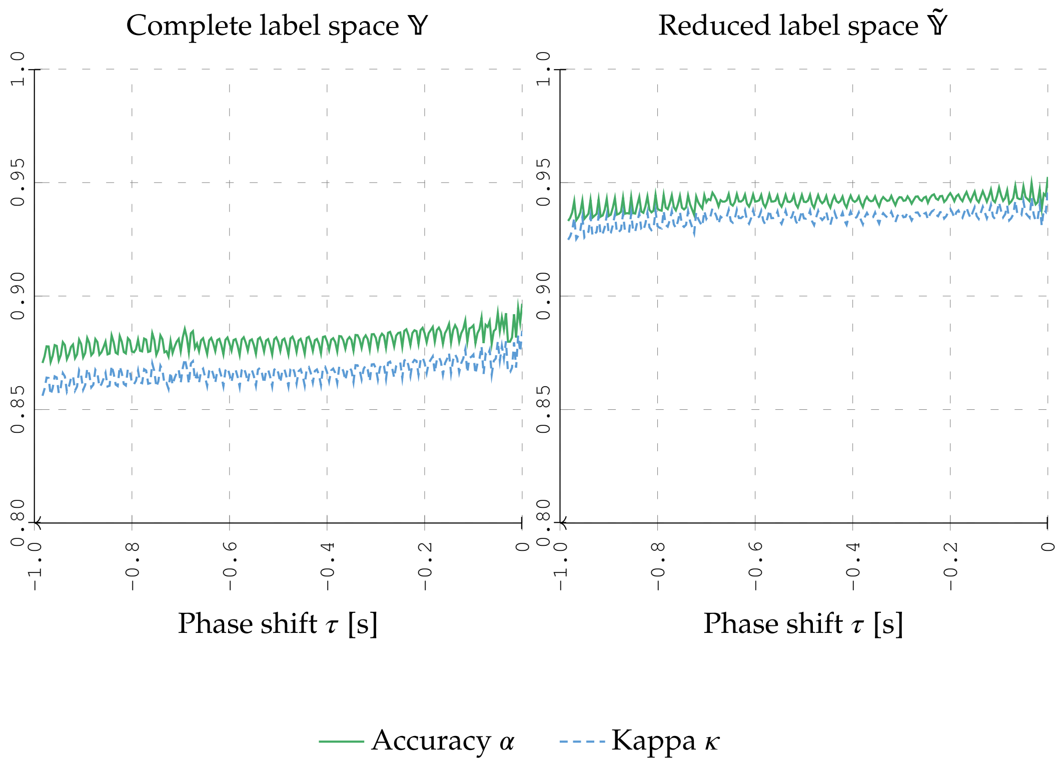


Figure 2.10.: Illustration of the translation invariance of the proposed model and it sensitivity to phase shift in load signatures. The model is evaluated on a phase-shifted variants $\mathscr{D}^{(\text{test})}(\tau)$ of the evaluation set $\mathscr{D}^{(\text{test})}$ in each cross-validation fold. The shift spans a 1-second window starting from the end of each measurement and going backward in time with steps of 125 samples such that $\tau = 0, 125, 250, \cdots, 30000$ samples. Observed are a slowly declining trend and minimal variation per period. Accounting for variations, the best recorded accuracy is $\alpha = 0.8892$ (± 0.0072) for the complete label space.

The second phenomena is the oscillatory behavior observed in Figure 2.10 which represents fluctuations in the observed performance within the same period (around 16.67ms). Variations in the accuracy measure within the same period are in average 0.72%. Our claim is that these variations represent a side-effect of the relaxed expansion of the training set proposed in Equation (2.12) for computational feasibility. Since the observed variations are minimal, we delegate further investigations on this corollary to future work and report our best load identification accuracy as 0.8892 ($\pm$0.00745) with a complete label space and 30 kHz sampling frequency.

Table 2.2.: The per-load F_1-score measure of our model compared to related work on the PLAID dataset [Gao et al. 2014]. First four loads are a compact fluorescent lamp (`CFL`), a fridge (`FR`), a microwave oven (`MC`), and an air conditioner (`AC`).

F_1-score	Gao et al. 2015	Baets et al. 2017a	Baptista et al. 2018	**ours**
`CFL`	**0.971**	0.956	0.909	0.698
`FR`	0.523	0.510	0.589	**0.969**
`MC`	**0.934**	0.931	0.870	0.740
`AC`	0.383	0.467	0.612	**0.926**
Hairdryer	0.784	0.798	**0.847**	0.741
Laptop	0.922	**0.979**	0.880	0.774
Vacuum	0.974	**0.979**	0.976	0.882
Bulb	0.840	0.806	0.848	**0.956**
Fan	0.689	0.601	0.542	**0.986**
Washer	0.698	0.688	0.806	**0.961**
Heater	0.000	0.822	0.719	**0.894**
macro-F_1-score	0.702	0.776	0.782	**0.866**
weighted-F_1-score	0.797	**0.824**	0.809	0.823
accuracy α	0.815	—	—	**0.889**

2.7.8. Related Work and Prior Art

Table 2.2 depicts the per-load F_1-score measure for the proposed model compared to related work on the same dataset, in addition to aggregate measures. Gao et al. [2015] utilized an RFT on binary VI-images while Baets et al. [2017a] and Baptista et al. [2018] proposed CNN architectures. The table shows a notable performance gain of our model in most load categories. While the class weighted F_1-score is almost similar to [Baets et al. 2017a], the ease of domain knowledge incorporation or modular architecture (as discussed in Section 2.5.3) is not equally possible in standard CNN architectures.

2.8. Conclusion and Future Work

In this chapter, we introduced a plug-level load monitoring model based on ensembles of small neural networks applied on raw, high-resolution current and voltage measurements with unsupervised, data-driven feature extraction. We evaluated our model on the real-work PLAID dataset with the best performance test achieving 0.8892 (±0.00745) of accuracy. The study shows the upper bound of a energy disaggregation in load identification assuming perfect disaggregation (emulated as plug-level monitoring) and at high sampling frequencies that most likely comprise most electricity-related features.

Additionally, we show the advantage of using the raw VI-waveforms as load signatures which was shown to promote identifiability of monitored loads. Further, the use of an unsupervised feature extractor (that is, a data-driven model) showed reliable learning pattern that remained more stable and robust to discretization errors in lower sampling rates.

It was observed that the performance notably degrades as the size of the training data is reduced. Therefore, one of the main topics of this work will be unsupervised and semi-supervised learning in energy monitoring and disaggregation (that is, Chapter 3 and Chapter 4) addressing situations of scarce or complete unavailability of training data.

While being able to reach state-of-art performance on end-use load identification in plug-level monitoring, we highlight in the sequel numerous directions in which this work can be extended in the future.

First, the one-vs-one ensemble results in a quadratic growth rate of the model size with respect to cardinality of the label space. While a one-vs-all ensemble is also possible and scales rather linearly, a problem of class imbalance will certainly arise. Future work may investigate more flexible model architectures that mitigate this scalability limitation.

Second, the chosen one-vs-one architecture was mostly motivated by the need for modular architectures and ease of domain knowledge incorporation. While standard multi-class neural net architecture do not feature such modularity by default, future work should focus on distilling such architectures.

Further, we proposed the one-vs-one ensemble architecture based on the hope of an abstaining behavior for out-of-distribution queries. For such a task, probabilistic models (e.g. BNNs) is more relevant and is another proposal for future work.

Chapter 3.

Unsupervised Energy Disaggregation

Obtaining clean labeled training data is oftentimes a tedious, expensive, and time-consuming stage in various data science applications, and that similarly applies to energy disaggregation. Training data acquisition for end-use energy disaggregation can itself become energy-inefficient [Kelly and Knottenbelt 2015b], requires laborious post-processing [Anderson et al. 2012], or becomes unreasonably invasive [Gupta et al. 2010]. Unlabeled measurements, on the other hand, are relatively much cheaper to obtain (even synthesize as in more recent works [Kelly and Knottenbelt 2015a, Chen et al. 2016, Henriet et al. 2018]) and require merely interfacing with the whole-hose power meter for acquisition and storage.

In this chapter, we introduce our work on unsupervised event-based energy disaggregation which is a multi-stage energy disaggregation framework relying primarily on abrupt changes in the aggregate signal in detecting and identifying end-use loads with minimal reliance on domain knowledge. While featuring an unsupervised learning scheme, the proposed framework additionally features a high level of interpretability. Moreover, the proposed framework showed notable robustness against minimal variations in hyper-parameters that, in turn, facilitated deployment in residential and commercial settings.

This chapter is organized as follows. Section 3.1 gives a brief overview of the general event-based unsupervised energy disaggregation framework. Section 3.2 states the addressed problem in this chapter. The following section, Section 3.3, briefly introduces two energy datasets that will be adopted for empirical validation throughout this chapter. The disaggregation framework is then detailed one stage at a time starting with event-detection in Section 3.4, feature extraction in Section 3.5, event-level clustering Section 3.6, and finally load recognition and inverse load profile reconstruction in Section 3.7. Finally, Section 3.10 concludes this chapter with final remarks and future work.

This work has been published by Barsim et al. [2013; 2014a;b], Barsim and Yang [2016], Wild et al. [2015] and formulations, phrasing, experiments, and results are partially adopted from our prior works. Additionally, part of this work was a contribution from an earlier work [Barsim 2013]. Finally, this work is open-sourced under: `https://github.com/karimpedia/tafSeel`.

3.1. Introduction

With the task of energy disaggregation defined (see Section 1.1), an unsupervised setting thereof is a setting in which the task is addressed with no reliance on externally labeled data neither in the form of sub-metered load profiles nor manually labeled load events. This is primarily motivated by the fact that acquiring labeled, not to mention clean, data in an adequate abundance is, in many applications, an expensive and laborious task. In contrast, unlabeled data can be acquired in ubiquitous amounts at relatively minimal costs compared to labeled data.

Even though an unsupervised disaggregation system will not be able to assign human-sensible load names or categories due to the loss of labeled data, it definitely assists in load profile reconstruction, which can then be easily labeled e.g. using a semi-supervised system as shall be discussed in Chapter 4.

An event-based setting for a disaggregation system is a setting in which loads' profiles are inferred primarily from the state-changes of end-use loads which are known commonly as *events*, hence the name. Such a disaggregation system relies mainly on two assumptions. The first is that loads reflect their state-change events in the form of abrupt changes in the aggregate load profile. The second assumption is that state-change events are temporally sparse compared to the sampling rate of the load profile. In other words, it is assumed that a single, or a limited number of, loads may change state in a short interval of time. This is sometimes referred to as the *switch continuity principle* (SCP) within the energy disaggregation community [Makonin 2016].

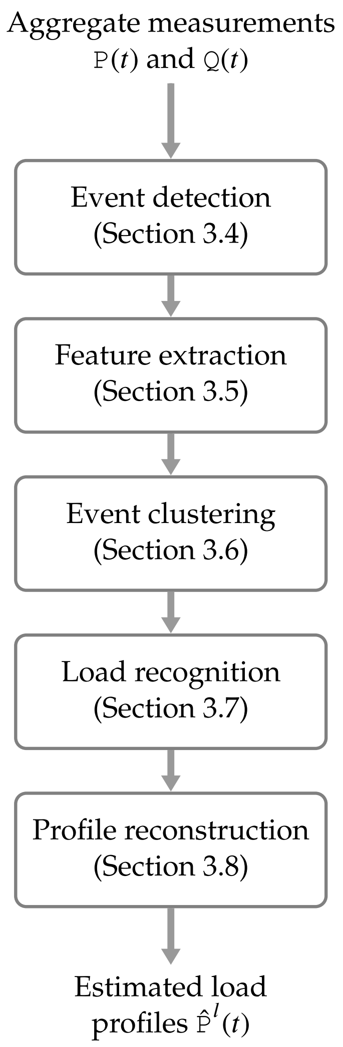

Aggregate measurements $\mathtt{P}(t)$ and $\mathtt{Q}(t)$
Event detection (Section 3.4)
Feature extraction (Section 3.5)
Event clustering (Section 3.6)
Load recognition (Section 3.7)
Profile reconstruction (Section 3.8)
Estimated load profiles $\hat{\mathtt{P}}^l(t)$

Figure 3.1.: A block diagram representing the standard event-based unsupervised energy disaggregation pipeline which will be adopted in this chapter.

A standard pipeline for an event-based energy disaggregation system comprises four stages as depicted in Figure 3.1. The first is an event-detection stage tasked with detecting abrupt changes in the aggregate load profile. This is then followed by a feature extraction stage responsible for extracting representative feature vectors from each detection to reliably distinguish between distinct loads in the monitored circuit. Particular to unsupervised disaggregation chains, the third stage is an event clustering stage applied to extracted features with the objective of clustering detections that are most likely to originate from the same load instance. A second objective of the event clustering stage is to identify falsely claimed event detections thus diminishing their impact on the following stages. The following stage, load recognition, is where load components are constructed from these event detections along with their clustering structures. From the this stage, load usage statistics (e.g. usage time, consumed energy, stage-changes ... etc) are inferred. Optionally, a time series segregated load profile may be also reconstructed in the last stage. Our proposed framework follows this standard pipeline, and these five stages are the main topic of Sections 3.4 to 3.8.

3.2. Problem Statement

The problem addressed in this chapter is stated as follows. *Given the aggregate load profile of a building, how much of the total energy consumption can an unsupervised disaggregation framework attribute to distinct end-use loads?*

We illustrated in the first chapter (Section 1.2.2) the overloading usage of the term *unsupervised disaggregation*, and explicated our definition thereof which matches the machine learning definition. Accordingly, we assume in this chapter that labeled data is unavailable or inaccessible. Additionally, we assume missing information about the number of loads in the target building, types or categories of these loads, nor any relevant characteristics (e.g. nominal power draw, usage times ... etc). We address this problem with a rule-based heuristic disaggregation pipeline that will be empirically validated on residential and commercial datasets, which will be introduced in the following section.

3.3. Evaluation datasets

The proposed framework has been empirically validated and evaluated at different stages of disaggregation on two real-world datasets. The first is a residential dataset developed and published specifically for event-based energy disaggregation research, and is acronymed by its developers as BLUED. The second is a commercial dataset that was acquired from the workplace of a company in Germany, whose name as well as various aspects of the dataset will remain anonymous.

3.3.1. The residential dataset

The building-level fully labeled energy disaggregation (BLUED) dataset [Anderson et al. 2012, Anderson 2014] is a ∼7-day long electrical energy dataset acquired from a residential building in Pittsburgh, Pennsylvania, in October 2011 that was, shortly afterwards, made public[1] to motivate and support energy disaggregation research. The monitored building is a single-family residence with

Remarkable in BLUED, and distinct from earlier energy datasets, is the accompanied labeling of each event (that is, state changes of individual end-use loads) throughout the whole monitoring period. The event-level labeling was acquired via a separate, plug-level[2] measurement system that ran in parallel to the aggregate measurement acquisition system. Missing, however, from this dataset is the sub-metered load profiles that would have of a notable value in evaluating the load profile reconstruction stage Section 3.8. For this reason, we evaluate our reconstructed load profiles on a commercial dataset that will be introduced shortly (see Section 3.3.2).

The monitored residence had a two-phase[3] split of loads (referred to hence forth as phase A and B). The dataset developers noted around 50 loads in the target residence but only a fraction was observed in that short period of acquisition. Worth mentioning as well is the notable imbalance between the two phases which is observable from the number of loads, and rate of state changes in each phase. Phase A comprised fewer number loads, and the majority of its events were generated by a single load, namely the fridge whereas phase B comprised a larger variety of load categories, more state-change events, and as a result more frequent violations of the switch continuity principle. Further details on these events will be given in Section 4.7.

The nominal grid frequency in the US is 60Hz, and BLUED's 12kHz raw current and voltage measurements were accompanied with the post-processed real and reactive

[1]`http://portoalegre.andrew.cmu.edu:88/BLUED/` at the time of this writing.

[2]In addition to plug-level measurement system, the authors devised feasible workarounds (e.g. electro-magnetic sensors) to monitor loads that were unattainable via a plug-socket connection.

[3]In fact, it is two 120V sources originating from a 240 single-phase distribution. For this work, we shall refer to these sources as *phases* (that is, phase A and B) as has been similarly adopted by BLUED authors [Anderson et al. 2012].

power signals at the grid frequency. These two signals represent the main input to our energy disaggregation framework.

3.3.2. The commercial dataset

The commercial dataset adopted for evaluation is a two-week dataset acquired from the workplace of a company in Berlin. Raw current and voltage measurements were acquired at 4 kHz rate, but post-processed to the real and reactive power signals at the grid frequency which is 50 Hz in Germany. The workplace similarly comprised at least 50 loads, but most were observed during the two-week period, and additionally comprised various load categories that are uncommon in a normal residence (e.g. vending machines, copy machines, elevators, multiple identical desktop machines, lighting groups ... etc).

The dataset was accompanied with the sub-metered[4] load profiles which enabled the evaluation of the later stages of our proposed framework, namely the inverse load profile reconstruction. Sub-metered channels comprised both real and reactive power signals sampled at a 1 Hz rate. The building had a 3-phase 240V distribution (referred to phase 1 to 3) and some loads were not restricted to a single phase.

3.4. Clustering-based Event Detection

The problem of change-point detection has been well studied and adopted in many signal processing applications [Basseville and Nikiforov 1993, Amini and Gallinari 2002, Liu et al. 2012]. In most of these applications, the informative segments of the signal are the stationary ones preceding and following the transient phase. However, for energy monitoring and disaggregation the transient phase represents a valuable load signature [Chang et al. 2010, Meziane et al. 2017, Leeb et al. 1995, Patel et al. 2007, Chen et al. 2013, Chang et al. 2008]. This raised the need for estimating the entire change interval (rather than change point) for a more reliable and stable extraction of load-representative features.

[4]In fact and due to resource limitations, some plug-level meters were attached to multiple loads. For instance, it was common to aggregate the whole desktop utilities (laptop, monitor, docking station, telephone, mobile charger ... etc) in one sub-metered channel.

In this section, we introduce a clustering-based framework for sequential detection of abrupt changes with accurate segmentation of the input signal into stationary and transient intervals. While we do not leverage transient signatures, we show that such segmentation leads to more robust and reliable feature extraction.

The main principle in clustering-based abrupt change detection is that stationary phases of a time series signal cluster together upon loosing their temporal dimension (that is, the finite sequence $\mathbf{X}_{a:b}$ of the d_x-dimensional discrete-time signal $\boldsymbol{x}(t)$ would form a cluster when projected on $\mathbb{R}^{d_x}$ if it is a stationary sequence). Similarly, transient time series segments are observed as outliers or scattered points when projected on their corresponding spatial dimensions $\mathbb{R}^{d_x}$ [Streubel and Yang 2012]. Figure 3.2 depicts few examples of this principle from the real and reactive power signals in a commercial building.

We leverage the reverse implication of this principle and state the following. If a finite sequence $\mathbf{X}_{a:b} = [\boldsymbol{x}(a+1), \boldsymbol{x}(a+2), \cdots, \boldsymbol{x}(b)]^\top$ does not show any clustering behavior when projected on its spatial dimensions $\mathbb{R}^{d_x}$, then it is unlikely that it contains any stationary sub-segments. Similarly, if $\mathbf{X}_{a:b}$ forms M clusters when projected on $\mathbb{R}^{d_x}$, then it is most likely to comprise at least[5] M stationary sub-segments. Evidently, a sequence that constitutes M stationary sub-segments comprises $M-1$ transient segments.

3.4.1. Event Model

Without loss of generality, we will primarily consider the T-length sequence $\mathbf{X}_{0:T} \in \mathbb{R}^{T \times d_x}$. Once projected[6] on the spatial dimensions $\mathbb{R}^{d_x}$ and with a suitable distance function $d(\cdot, \cdot)$, a spatial clustering algorithm can be applied to the projected data points. In our experiments, we utilize the scaled Euclidean distance function defined as

$$d\big(\boldsymbol{x}(t), \boldsymbol{x}(t')\big) = \Big(\big(\boldsymbol{x}(t) - \boldsymbol{x}(t')\big)^\top \boldsymbol{\Sigma}^{-1} \big(\boldsymbol{x}(t) - \boldsymbol{x}(t')\big)\Big)^{1/2} \tag{3.1}$$

[5]Multiple stationary sub-segments with the same nominal value are projected to the same cluster even if they appear in two disjoint time intervals.

[6]That is, mapping the sequence $\mathbf{X}_{0:T} \in \mathbb{R}^{T \times d_x}$ to its spatial subspace $\mathbb{R}^{d_x}$ parallel to the time axis, which is achievable by simply discarding the temporal index.

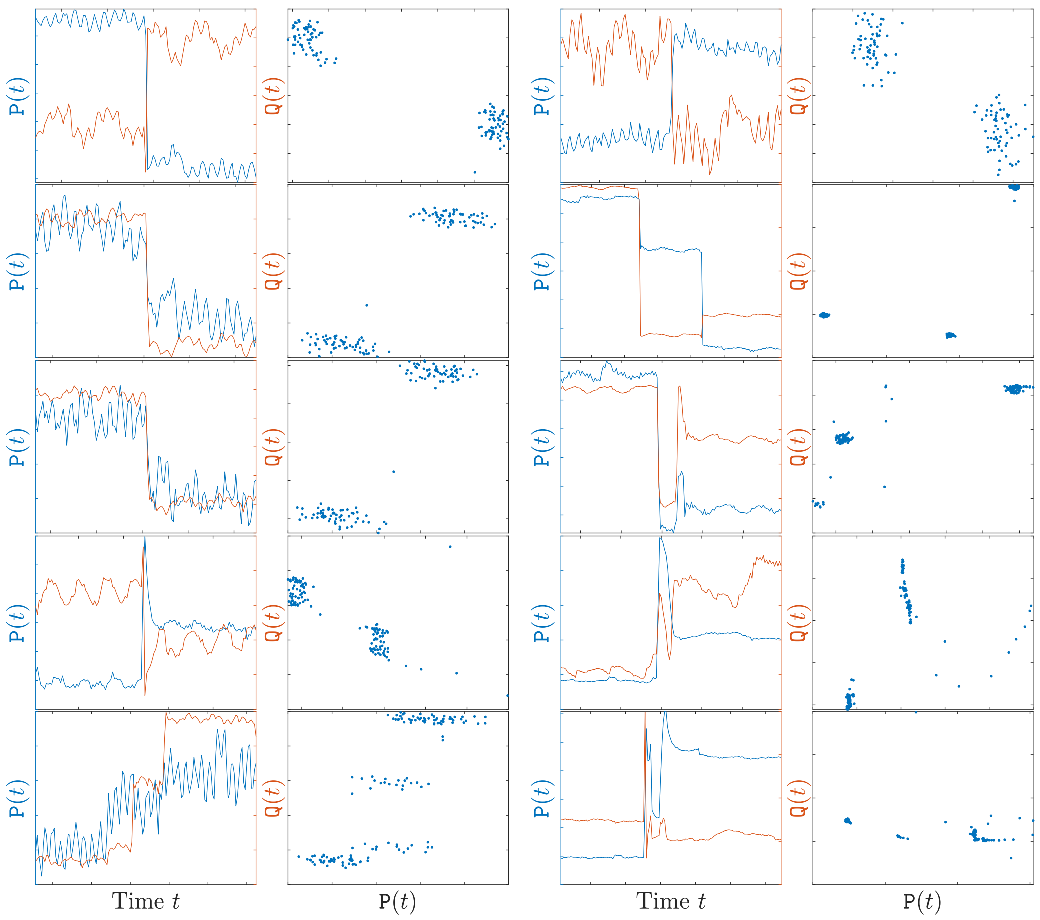


Figure 3.2.: Illustration of the clustering principle leveraged in the proposed clustering-based event detection. Each pair of adjacent sub-plots show the real power (blue) and the reactive power (orange) signals in time series (left) and its image on $\mathbb{R}^2$ (right). Observe that stationary phases appear as clusters in the $\mathbb{R}^2$ projection, while transient phases are represented by scattered outliers. Axes scales were discarded for convenience.

where $\mathbf{\Sigma}$ is the diagonal scale matrix given by

$$\mathbf{\Sigma} = \mathrm{diag}(\sigma_1^2,\, \sigma_2^2,\, \cdots, \sigma_{d_x}^2) \tag{3.2}$$

with diag(·) being the diagonal matrix denotation. Scale parameters σ_1, σ_2, $\cdots$ represent the first degree of freedom in the proposed event detection algorithm and are interpreted as the minimum change to detect in each dimension. For the real and reactive aggregate power signals $\boldsymbol{x}(t) = [\mathrm{P}(t),\ \mathrm{Q}(t)]^\top$, scale parameters are chosen to have $\sigma_1 = 15$ Watt and $\sigma_2 = 15$ VAR, respectively, in all our experiments.

For the clustering algorithm, we utilize the <u>d</u>ensity <u>b</u>ased <u>s</u>patial <u>c</u>lustering for <u>a</u>pplications with <u>n</u>oise (DBSCAN) algorithm [Ester et al. 1996] which returns three estimates, namely, an estimated number of clusters M, a mapping from the given sequence to cluster labels, and a second mapping from the sequence to the connectivity or clustering level — all will be detailed in the following.

DBSCAN partitions a given set of data points (e.g. the data points projected from the sequence $\mathbf{X}_{0:T}$) to mutually exclusive cluster sets. Worth mentioning here is that the number of clusters $M \geqslant 0$ is estimated by the DBSCAN algorithm and can be zero-valued if, for example, the observed data points do not form any dense regions (that is, cluster-less data).

Stated more formally, given the sequence $\mathbf{X}_{0:T}$ and the distance function in Equation (3.1), DBSCAN estimates the number of clusters M and builds thereupon the clustering label space $\mathbb{Y} = \{y_m\}_{m=0}^{M}$ and estimates a mapping from each of the sequence instances $\boldsymbol{x}(t)$ to an assigned cluster label $y(t) \in \mathbb{Y}$ for that instance $\boldsymbol{x}(t) \mapsto y(t)$. This results in a cluster sequence denoted by $\boldsymbol{y}_{0:T} = [y(1),\ y(2),\ \cdots,\ y(T)]^\top \in \mathbb{Y}^T$. Note that the label y_0 denotes a cluster-less set, that is, the set of samples which do not fit any of the retrieved clusters, and will be denoted oftentimes as *noisy samples* or *outliers*.

Should we denote by $\mathcal{Y}_m$ the set of sequence instances assigned the cluster label y_m such that

$$\mathcal{Y}_m = \left\{ \boldsymbol{x}(t) \in \mathbf{X}_{0:T} \mid y(t) \ = \ y_m \right\} \tag{3.3}$$

(where we have abused the notation $\boldsymbol{x}(t) \in \mathbf{X}_{0:T}$ to denote the membership of $\boldsymbol{x}(t)$ to the image of $\mathbf{X}_{0:T}$ in $\mathbb{R}^{d_x}$ given by $\{\boldsymbol{x}(t) \mid 1 \leqslant t \leqslant T\}$), then each cluster set $\mathcal{Y}_m$ is a spatially dense collection of points which is likely to represent a stationary segment in the sequence $\mathbf{X}_{0:T}$. An exception yet is the noise cluster denoted by y_0 whose sample set $\mathcal{Y}_0$ are indeed likely to be measurement noise but in our application may additionally represent the transient segment of a detected event in $\mathbf{X}_{0:T}$. For DBSCAN, all cluster sets are mutually exclusive $\mathcal{Y}_m \cap \mathcal{Y}_{m'} = \varnothing$ if $m \neq m'$ and non-empty except for the noise cluster $\mathcal{Y}_m \neq \varnothing\ \forall m > 0$.

Finally, DBSCAN distinguishes between two-levels of clustered samples, namely, cores and borders. Loosely speaking, core samples are those samples embedded in a dense region (that is, samples with a dense neighborhood) while borders are the neighborhood of core samples that do not retain a dense neighborhood by themselves. This is not of a paramount relevance to our detection algorithm, but can of course be leveraged (e.g. for reliable and more stable feature extraction). The outcome is another binary sequence $\mathbf{c}_{0:T} = [c(1), c(2), \cdots, c(T)]^{\top} \in \{0, 1\}^{\top}$ where $c(t)$ is a binary indicator on whether or not the sequence instance $\boldsymbol{x}(t)$ possess a dense neighborhood.

In addition to the distance function, DBSCAN takes a single[7] parameter known commonly as the minimum number of points `minPts` to identify a core sample (that is, a sample with a sufficient number of data points in its neighborhood) which represents the second degree of freedom in the proposed event detection algorithm. The `minPts` parameter is interpreted as the minimum length of a stationary segment and is set in all our experiments to 1 second (that is, 60 points in $F_g = 60$ Hz signals, and 50 points when $F_g = 50$ Hz[8]). This gives a limit to simultaneous events detection (that is, detection of events occurring in close temporal proximity) and an intuitive adjustment parameter controlling the trade-off between retrieval (of simultaneous events) and relevance (e.g. false detections resulting from over-segmenting long transients).

The first requirement for a sequence $\mathbf{X}_{0:T}$ to contain an event is that its image on $\mathbb{R}^{d_x}$ forms at least two clusters $M \geqslant 2$ which is a necessary (but insufficient) condition. For sufficiency, we impose two more constraints for which we first introduce the following properties of a cluster.

Let $\mathcal{t}_m$ denote the set of time indices exhibiting a membership to the m^{th} cluster such that

$$\mathcal{t}_m = \left\{ \; t \in \mathbb{N}_{\leqslant T} \mid y(t) = y_m \; \right\} \tag{3.4}$$

which leads to the concept of cluster interval denoted by $[\![y_m]\!]$ and defined as

$$[\![y_m]\!] = \left[\!\!\left[\min \mathcal{t}_m, \; \max \mathcal{t}_m \right]\!\!\right] \tag{3.5}$$

[7]In fact, DBSCAN takes two parameters; `minPts` and neighborhood threshold ϵ [Ester et al. 1996]. However, the second parameter is embedded in our case in the distance function as the σ_i parameters and hence is no longer explicitly required by the DBSCAN algorithm.

[8]Assuming that signals are sampled at the rate of the grid frequency.

where $[\![a,\ b]\!]$ denotes an integer interval from the inclusive endpoints a to b given by $\{z \in \mathbb{Z} \mid a \leqslant z \leqslant b\}$ and $\min \mathcal{X}$ is the minimum element in the set $\mathcal{X}$, and likewise for $\max \mathcal{X}$. We then define cluster cardinality, temporal locality, and overlap with another cluster as follows.

Cardinality of a cluster label y_m is the number of data samples assigned to that cluster and is given by $|\,\mathcal{t}_m\,|$.

Locality of a cluster y_m is a measure of how dense the cluster samples are in time (that is, considering the distance $\tilde{d}(\boldsymbol{x}(t), \boldsymbol{x}(t')) = |t - t'|$). It is denoted by $\mathrm{loc}(y_m)$ and is estimated as the ratio between cluster size and it temporal length

$$\mathrm{loc}(y_m) = \frac{|\,\mathcal{t}_m\,|}{\mathrm{len}\left([\![y_m]\!]\right)} \quad \in (0, 1] \tag{3.6}$$

where the $\mathrm{len}([\![a,\ b]\!])$ denotes the length of the integer interval $[\![a,\ b]\!]$ which is given by $(b - a + 1)$. A temporally local cluster $\mathrm{loc}(y_m) \rightarrow 1$ indicates that this cluster is formed out of adjacent instances $\boldsymbol{x}(t)$ in the considered time series sequence $\mathbf{X}_{0:T}$.

Overlap between two clusters y_m and $y_{m'}$ where $m \neq m'$ and $m, m' > 0$ is defined as the intersection of their corresponding temporal intervals which is denoted by $[\![y_m]\!] \cap [\![y_{m'}]\!]$.

A sequence $\mathbf{X}_{0:T}$ represents an abrupt change event if its image on $\mathbb{R}^{d_x}$ comprises two temporally local clusters y_m and $y_{m'}$ such that

$$\mathrm{loc}(y_m),\ \mathrm{loc}(y_{m'}) \geqslant \vartheta_{\mathrm{loc}} \tag{3.7}$$

and with minimal overlap such that

$$\frac{[\![y_m]\!] \cap [\![y_{m'}]\!]}{\texttt{minPts}} \leqslant \vartheta_{\cap} \tag{3.8}$$

The two parameters ϑ_{loc} and $\vartheta_{\cap}$ are yet two more degrees of freedom in our event model where the first is interpreted as the permitted rate of measurement outliers during the stationary sub-segments, and the second permits more flexibility in the transient phase (the higher, the more flexible). In our experiments these two parameters were chosen to $\vartheta_{\mathrm{loc}} = 25\%$ and $\vartheta_{\cap} = 50\%$ which were found to be reasonable values for both residential and commercial aggregate and segregate load profiles.

Algorithm 3.1 The clustering-based event model queried for an event in the cluster sequence $\boldsymbol{y}_{0:T}$.

```
Input: minPts: minimum length of a stationary segment
Input: $\vartheta_{\text{loc}}$: Locality threshold
Input: $\vartheta_{\cap}$: Overlap threshold
1: procedure QUERYEVENT($\boldsymbol{y}_{a:b}$, $M$)
2:     if $M < 2$ then
3:         return False
4:     for each pair $(m, m') \in \mathbb{N}^2_{\leqslant M}$ do
5:         if $\text{loc}(y_m) \leqslant \vartheta_{\text{loc}}$ and $\text{loc}(y_{m'}) \leqslant \vartheta_{\text{loc}}$ then
6:             if $[\![y_m]\!] \cap [\![y_{m'}]\!] \leqslant \vartheta_{\cap} \cdot$ minPts then
7:                 return True
8:     return False
```

Algorithm 3.1 illustrates the conditions tested for each query for an event in a given time series sequence. In a nutshell, whenever a sequence $\mathbf{X}_{0:T}$ is queried for the existence of an abrupt change event, then 1) construct the image of the sequence on its spatial dimensions $\mathbb{R}^{d_x}$, 2) apply a clustering algorithm (for example, DBSCAN) to estimate the number of clusters in the sequence image on $\mathbb{R}^{d_x}$ and construct the cluster sequence $\boldsymbol{y}_{0:T}$, 3) amongst temporally local clusters $\text{loc}(y_m) \geqslant \vartheta_{\text{loc}}$, 4) retrieve the pair of clustered sub-segments with minimal overlap $[\![y_m]\!] \cap [\![y_{m'}]\!] \leqslant \vartheta_{\cap} \cdot \texttt{minPts}$. If at least one pair of such clusters exist, the sequence $\mathbf{X}_{0:T}$ is likely to represent an event.

The event model parameters are the scale parameters $\sigma_1, \sigma_2 \in \mathbb{R}_{>0}$, the minimum stationary segment length `minPts` $\in \mathbb{N}$, the rate of measurement outliers in stationary phases $\vartheta_{\text{loc}} \in [0, 1]$, and the model's flexibility[9] during the transient phase $\vartheta_{\cap} \in \mathbb{R}_{>0}$.

3.4.2. Detection Algorithm

The algorithm for extracting event sub-sequences from the signal $\boldsymbol{x}(t)$ that are matching the event model introduced earlier can now be stated as follows. Starting from a window of width $2 \cdot$ `minPts` (that is, 2 seconds and is the minimum window to

[9]While this is a vague interpretation, it in fact means that the transient phase can have an arbitrarily shaped profile with some instances $\boldsymbol{x}(t)$ possibly coinciding with either of the preceding or the following stationary phases.

Algorithm 3.2 The proposed clustering-based event detection algorithm. The queried sequence $\tilde{\mathbf{X}}_{a:b}$ is obtained from the scaled aggregate signal $\tilde{\boldsymbol{x}}(t)$, the QueryEvent procedure is defined in Algorithm 3.1, and the DBSCAN procedure is the canonical algorithm from [Ester et al. 1996]. The algorithm only returns the first occurrence of an abrupt change event and has to be repeated starting from e.g. the right endpoint b.

```
Input: minPts: minimum length of a stationary segment
Input: $\boldsymbol{\Sigma}$: Scale
 1: procedure DETECTEVENT($\boldsymbol{x}(t)$)
 2:     $\tilde{\boldsymbol{x}}(t) \leftarrow \boldsymbol{x}(t)\boldsymbol{\Sigma}^{\frac{1}{2}}$                ▷ Scale
 3:     $a \leftarrow 0$                                  ▷ Left endpoint
 4:
 5:     $b \leftarrow a + (2 \cdot$ minPts $- 1)$             ▷ Right endpoint
 6:     repeat                                    ▷ Forward detection
 7:         $b \leftarrow b + 1$
 8:         $\boldsymbol{y}_{a:b}, \boldsymbol{c}_{a:b}, \boldsymbol{M} \leftarrow$ DBSCAN($\tilde{\mathbf{X}}_{a:b}$, minPts, $\epsilon = 1$)
 9:         event $\leftarrow$ QueryEvent($\boldsymbol{y}_{a:b}$, $\boldsymbol{M}$)
10:     until event detected
11:
12:     $a \leftarrow b - (2 \cdot$ minPts $- 1)$
13:     repeat                                    ▷ Backward refinement
14:         $a \leftarrow a - 1$
15:         $\boldsymbol{y}_{a:b}, \boldsymbol{c}_{a:b}, \boldsymbol{M} \leftarrow$ DBSCAN($\tilde{\mathbf{X}}_{a:b}$, minPts, $\epsilon = 1$)
16:         event $\leftarrow$ QueryEvent($\boldsymbol{y}_{a:b}$, $\boldsymbol{M}$)
17:     until event detected
18:
19:     return ; $a$, $b$
```

detect an event in our experiments), the window iteratively expands from the right endpoint and the enclosed sequence is repeatedly queried for the existence of an event based on the model constraints discussed earlier. Once an event is detected, it retracts iteratively from the left endpoint to estimate the smallest window that comprises the detected event. Equivalently, the same objective (that is, estimating the minimum window comprising the event) is achieved upon expanding to the left starting from the right endpoint after the forward detection. The latter approach is expected to result in computational savings especially in the case of rare (aka temporally sparse) events, and is the one adopted in our experiments.

Algorithm 3.2 illustrates the proposed event detection algorithm. The detection algorithm comprises a forward detection loop followed by a refinement step once an event is detected. For a practical implementation, it is noted that the canonical DBSCAN algorithm is of a worst-case quadratic complexity $O(N^2)$ in both time and space and is, therefore, very demanding to re-apply every instance t in the given signal $\boldsymbol{x}(t)$. Therefore, we adopt a modified implementation depicted in Algorithm 3.3 with an expansion step $T_s \in \mathbb{N}$ and an upper bound on the monitoring window $T_w \in \mathbb{N}$. Greedy values for the canonical version are a unit step $T_s = 1$ and an infinitely wide monitoring $T_w = \infty$. In all our experiments for both residential and commercial settings, we used a half-second expansion or sliding step $T_s = F_g \times 0.5\text{s}$ and a one minute monitoring window $T_w = F_g \times 60\text{s}$.

Figure 3.3 depicts an illustrative example of the proposed sequential clustering-based event detection algorithm (Algorithm 3.3) applied on a 3-minute sequence of phase B of the BLUED dataset. The figure comprises three rows of sub-figures (top, middle, and bottom) to represent progression of the event detection algorithm in time. In each row, the real and reactive power signals (left), the current detection window (green shaded area), and the PQ-plane showing the image of power signals on $\mathbb{R}^2$ (right) are depicted. In the PQ-plane, the image of the whole 3-minute sequence is depicted in grayed out scatter points, while the sub-sequence under the detection window is highlighted in red. Moreover, an arrow above each shaded detection window explicates the direction in which the window is currently expanding. Detection starts with a suitably narrow window as shown on the top row of figures. At this stage, it is known that the sub-sequence under the detection window comprises at most a single cluster (shown on the highlighted red samples PQ-plane). Detection then proceeds by expanding its window forward in time (middle row of sub-figures) until a sub-sequence matches the event model and a detection is declared. In this case, the PQ-plane comprises at least two clusters. In its last step, the detection window retracts to the minimal length anchored at its end-point such that the detection flag is cleared (e.g. a window that is `minPts` in length or shorter, and this is not shown in the figure) and expands backward in time until a match is declared (bottom row of sub-figures). With this last step, the minimal length sub-sequence matching the event model is estimated. The algorithm further proceeds to the rest of the signal by re-initializing the whole procedure post-transient the currently detected event.

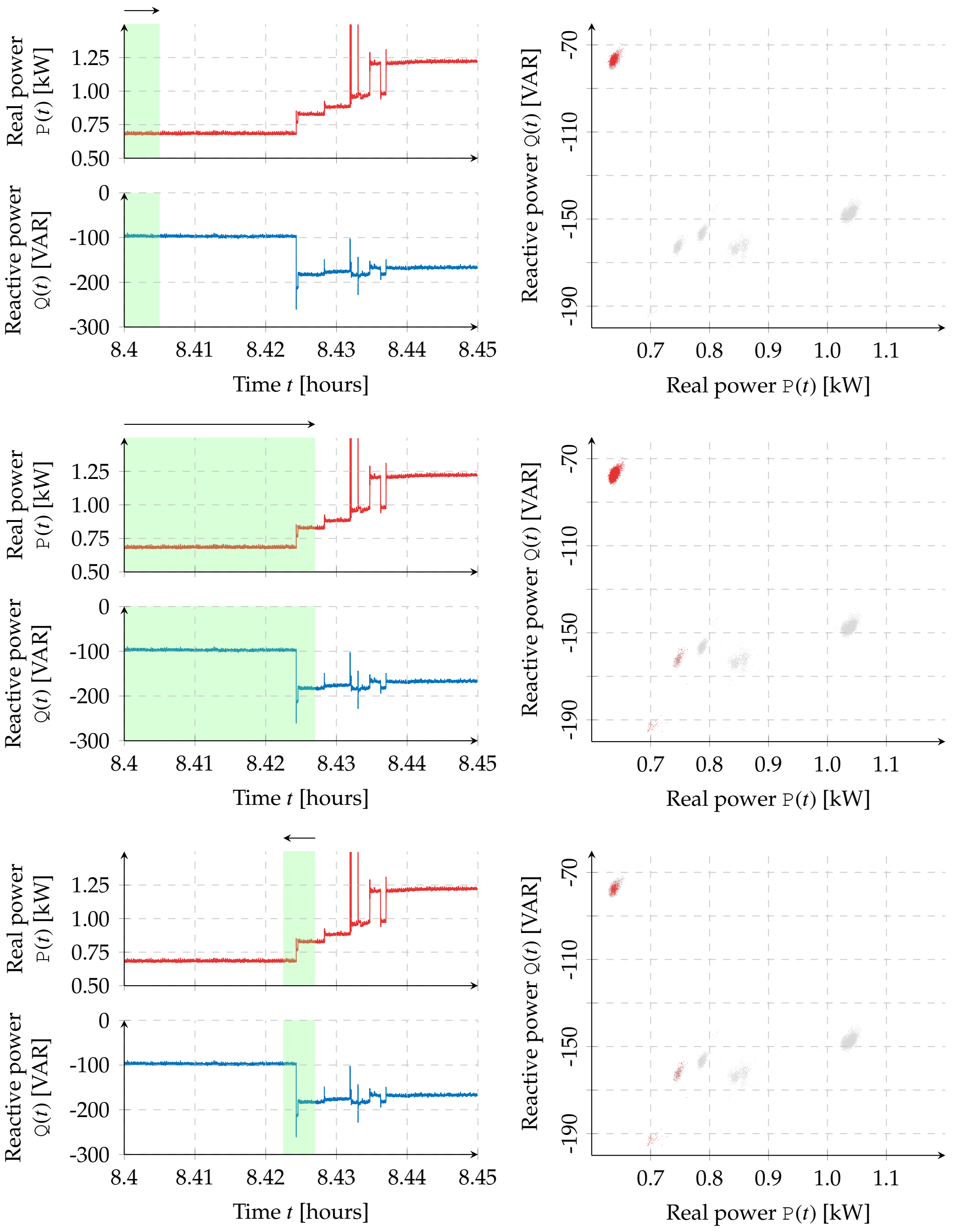


Figure 3.3.: A visual illustration of the three steps of the detection algorithm (e.g Algorithm 3.3) applied to a 3-minute window from phase B of the BLUED dataset with steps top) expansion (or forward detection), middle) detection (that is, event model match), and bottom) backward refinement. See. detailed description inline.

Algorithm 3.3 A less greedy but more computationally efficient variant of the detection algorithm depicted in Algorithm 3.2. Additional parameters are the maximum window size T_w and the sliding or expansion step T_s. Similarly, the algorithm only returns the first occurrence of an abrupt change event and has to be repeated starting from e.g. the right endpoint b. The function max(a, b) returns the maximum of the scalars a and b.

```
Input: minPts: minimum length of a stationary segment
Input: $\mathbf{\Sigma}$: Scale
Input: $T_w$: Maximum window size
Input: $T_s$: Sliding or expansion step
 1: procedure DetectEvent($\boldsymbol{x}(t)$)
 2:     $\tilde{\boldsymbol{x}}(t) \leftarrow \boldsymbol{x}(t)\mathbf{\Sigma}^{\frac{1}{2}}$                          ▷ Scale
 3:     $a \leftarrow 0$                                            ▷ Left endpoint
 4:
 5:     $b \leftarrow a + (2 \cdot \texttt{minPts} - T_s)$                      ▷ Right endpoint
 6:     repeat                                          ▷ Forward detection
 7:         $b \leftarrow b + T_s$
 8:         $a \leftarrow \max(a, b - T_w)$
 9:         $\boldsymbol{y}_{a:b}, \boldsymbol{c}_{a:b}, \boldsymbol{M} \leftarrow$ DBSCAN($\tilde{\mathbf{X}}_{a:b}$, minPts, $\epsilon = 1$)
10:         event ← QueryEvent($\boldsymbol{y}_{a:b}$, $\boldsymbol{M}$)
11:     until event detected
12:
13:     $a \leftarrow b - (2 \cdot \texttt{minPts} - 1)$
14:     repeat                                       ▷ Backward refinement
15:         $a \leftarrow a - 1$
16:         $\boldsymbol{y}_{a:b}, \boldsymbol{c}_{a:b}, \boldsymbol{M} \leftarrow$ DBSCAN($\tilde{\mathbf{X}}_{a:b}$, minPts, $\epsilon = 1$)
17:         event ← QueryEvent($\boldsymbol{y}_{a:b}$, $\boldsymbol{M}$)
18:     until event detected
19:
20:     return ; $a$, $b$
```

3.4.3. Empirical Validation

Since BLUED was developed primarily for evaluation of event-based energy disaggregation systems, we first report our detection results on BLUED's labeled events.

Table 3.1 shows a comparison between the generalized likelihood ratio (GLR) detector [Anderson et al. 2012], an event detector based on kernel Fisher discriminant analysis (KFDA)[10], a clustering-based event detector based on the mean-shift [Fuku-

[10]The work of a student under my supervision.

Table 3.1.: The per-phase detection measures (precision, recall and F_1-score) on each phase of the residential BLUED [Anderson et al. 2012] dataset.

		Anderson et al. 2012	Wild et al. 2015	Barsim et al. 2014a	**ours**
Phase A	recall	0.9816	**0.9878**	0.9841	0.9801
	precision	0.9794	**0.9966**	0.9943	0.9833
	F_1-score	0.9805	**0.9922**	0.9892	0.9817
Phase B	recall	0.7040	0.9217	0.7048	**0.9425**
	precision	0.8729	0.8632	**0.8897**	0.6963
	F_1-score	0.7794	**0.8915**	0.7865	0.8009

naga and Hostetler 1975] clustering algorithm [Barsim et al. 2014a], and the proposed DBSCAN-based event detector. The last three approaches represent our main contribution to abrupt change segmentation for reliable feature extraction.

Figure 3.4 depicts the event-detection results on the power signals of a small number of loads measured from a commercial building in Berlin at a sampling frequency of 1 Hz along with extracted power change features of all detected events. Further results are shown in Appendix A.

3.5. Feature Extraction

In this work, we primarily consider steady-state features, namely, the abrupt step changes in the real `P` and reactive `Q` power signals. We intentionally simplify the feature extraction stage in order to emphasize the effectiveness of the event segmentation performed in the prior stage. For each detected event we extract the step change in the real and reactive power signals, denoted `dP` and `dQ` respectively, from the change in the stationary segments preceding and following the segmented transient.

Let $\mathbf{X}_{0:T}$ denote a sequence of the aggregate signals the comprises an abrupt change event and, hence, forms two temporally local clusters $\mathcal{Y}_1$ and $\mathcal{Y}_2$. Assuming $\mathcal{Y}_1$ represents the leading stationary sub-sequence, then the event features are estimated

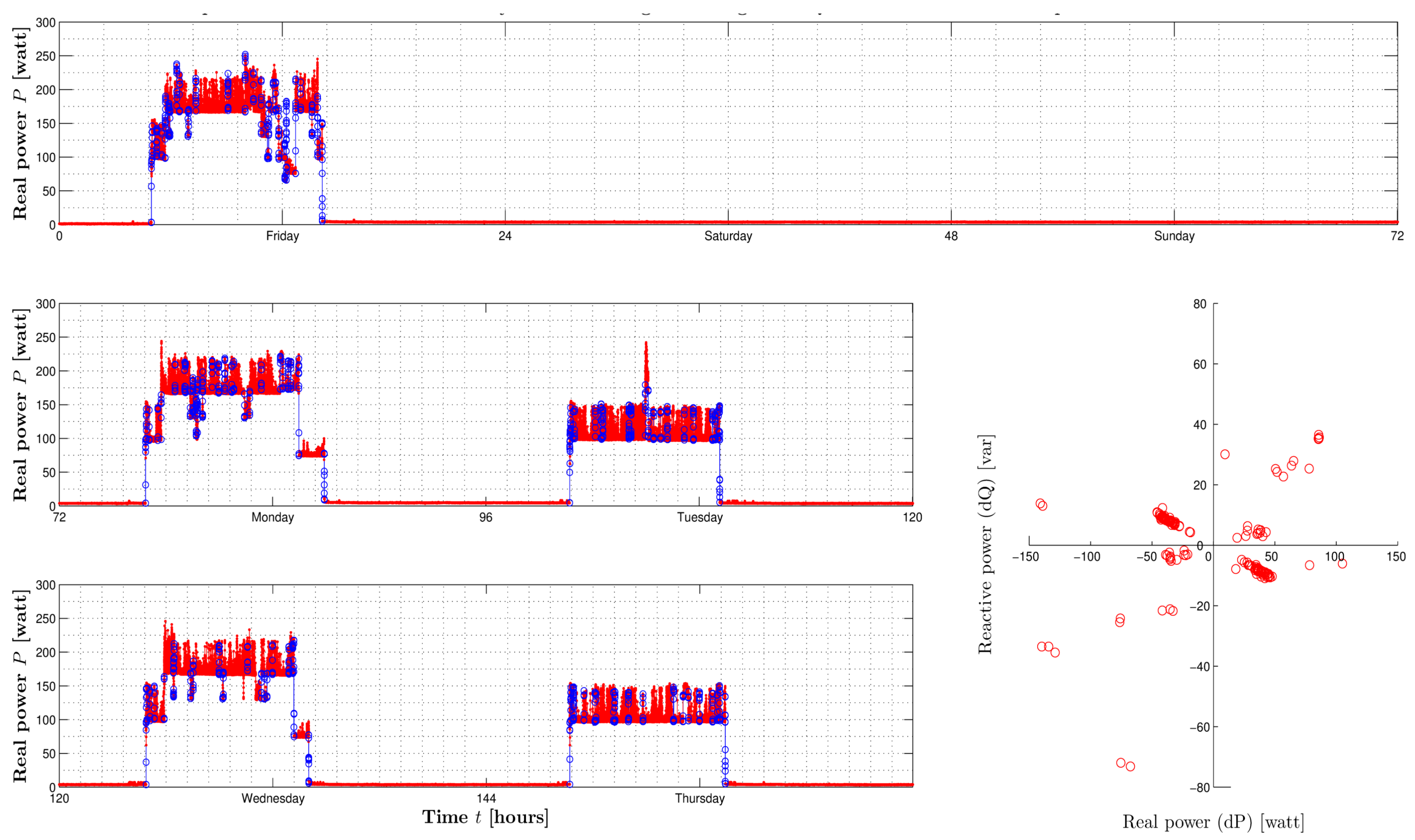


Figure 3.4.: DBSCAN-based event detection applied to 1 Hz power signals of a small number of loads from commercial building in Berlin. Red curves are the power signals while blue markers represent detected events. The lower right figure shows the power change features of detected events.

simply by the difference

$$\mathbf{d}\boldsymbol{x} = \begin{bmatrix} \mathtt{dP} \\ \mathtt{dQ} \end{bmatrix} = \frac{1}{|\mathcal{Y}_2|} \sum_{\boldsymbol{x}(t) \in \mathcal{Y}_2} \boldsymbol{x}(t) - \frac{1}{|\mathcal{Y}_1|} \sum_{\boldsymbol{x}(t) \in \mathcal{Y}_1} \boldsymbol{x}(t) \tag{3.9}$$

For a reliable, more robust, and noise-aware feature extraction, the secondary output of DBSCAN (that is, the clustering level sequence $\boldsymbol{c}_{0:T}$) can be leveraged in limiting the estimation of the cluster means (that is, each term of the right expression in Equation (3.9)) to the core samples. For that we define

$$\mathcal{Y}_m^{(\text{core})} = \{\boldsymbol{x}(t) \,|\, y(t) = y_m \text{ and } c(t) = 1\} \tag{3.10}$$

and feature extraction remains as defined in Equation (3.9) with the cluster sets $\mathcal{Y}_1$ and $\mathcal{Y}_2$ replaced with $\mathcal{Y}_1^{(\text{core})}$ and $\mathcal{Y}_2^{(\text{core})}$, respectively. In words, step changes are estimated based on the difference between cluster core representatives.

Additionally, we distinguish between switch-*on* and switch-*off* events based on the change in real power and associate each feature vector with its time of occurrence t (defined as the last sample in the leading cluster prior to the overlap). To that end, we split the data set of detected events denoted by $\mathcal{D} = \{(\mathbf{d}\boldsymbol{x}^{(n)}, t^{(n)})\}_{n=1}^{N}$, where N is the number of detected events, to two subsets $\mathcal{D}^{(\text{on})}$ and $\mathcal{D}^{(\text{off})}$ for switch-*on* and -*off* events, respectively.

For notational convenience, we will overload the symbol $\boldsymbol{x}$ to denote henceforth the feature vector of each detected event such that the dataset $\mathcal{D} = \{(\boldsymbol{x}^{(n)} \equiv \mathbf{d}\boldsymbol{x}^{(n)}, t^{(n)})\}_{n=1}^{N}$ comprises feature vectors of N detected events along with their time-of-occurrence $t^{(n)}$. Moreover, we will oftentimes drop the event index superscript $\cdot^{(n)}$.

Empirical validation of the robustness of extracted feature vectors will be illustrated via their effect on the following stage, that is, event-clustering. In Section 3.6.1, we demonstrate that extracted features exhibit more stable clustering structures with minimal intra-cluster variations while featuring maximal inter-cluster separation.

3.6. Event Clustering

The task of the event clustering stage is to group similar events based on the extracted event-based features into groups of high within-cluster similarity and wide across-cluster separation. In this section, we first review some of the challenges faced at this stage, and then detail our proposed approach.

Event clustering in a purely unsupervised energy disaggregation chain faces numerous challenges such as the a priori unknown number of loads nor states per load and, it turn, number of clusters, the high imbalance across cluster densities, variations within each cluster, and outliers resulting from false detections, to name a few.

Because neither the number of underlying loads nor the total number of distinct state transitions per load is known in advance, the clustering algorithm is additionally tasked with estimating the number of clusters. Additionally, and even though it is reasonable to assume that data samples can be fitted to an underlying generative model or tend to have specific spatial characteristics, we still assume that this information is also not available to the addressed unsupervised approach and, hence, the clustering algorithm should be robust to arbitrarily shaped clusters. Furthermore, different loads exhibit varying number of event generation (for example, thermostatically controlled loads are expected to result in many more events than lighting circuits). This is besides the challenge that the monitored circuit exhibit loads from few tens of Watts to kiloWatt loads and no preference is given to either in our approach.

The input to this stage is two sets of event-based features compiled in the set $\mathcal{D} = \{(\boldsymbol{x}^{(n)}, t^{(n)})\}_{n=1}^{N}$. Of most concern at this stage is the event features $\boldsymbol{x} \in \mathbb{R}^{d_x}$, and for that we consider a dataset of d_x-dimensional event-based features denoted by $\mathcal{X} = \{\boldsymbol{x}^{(n)}\}_{n=1}^{N}$ where N is the number of detected events.

To address the wide scale between small miscellaneous appliances and major loads, extracted features are transformed to the logarithmic domain using the function

$$\tilde{x} = f(x) = \operatorname{sgn}(x) \cdot \log\left(|x| + \beta\right) \tag{3.11}$$

where $\beta > 1$ is a shift parameter for numerical consideration (chosen in our case to be $\beta = 10$), $\log(\cdot)$ is the logarithmic function, $\mathrm{sgn}(\cdot)$ is the sign function, and the transformation $f(x)$ is applied element-wise in the multivariate case.

With the transformed feature vectors $\tilde{\boldsymbol{x}}$ from Equation (3.11) and the standard Euclidean distance $d(\boldsymbol{x}, \boldsymbol{x}') = ((\boldsymbol{x} - \boldsymbol{x}')^\top(\boldsymbol{x} - \boldsymbol{x}'))^{\frac{1}{2}}$, we utilize the DBSCAN clustering algorithm [Ester et al. 1996]. The result (as in Section 3.6) is an estimation of the number of clusters denoted by[11] M and partitioning of the dataset $\mathcal{X}$ into M mutually exclusive clusters $\mathcal{Y}_m \subseteq \mathcal{X}\ \forall m$ in addition to the noise cluster $\mathcal{Y}_0$ comprising outliers which are most likely to be false detections.

The challenges of unknown number of clusters and cluster distribution is addressed with the choice of the DBSCAN clustering algorithm which is known to be suitable for arbitrarily shaped clusters and features an implicit estimation of the total number of clusters. DBSCAN additionally addresses outliers resulting from false detections through its notion of noise. Finally, the wide feature scale is addressed using the logarithmic transformation proposed in Equation (3.11). DBSCAN, however, remains valuerable to imbalance in cluster densities and this addressed in our case through longer monitoring periods (around 2 weeks in our experiments) to give real events sufficient time to accumulate and form dense regions. This remains an open problem from our work and consensus clustering [Aggarwal and Reddy 2013] is a promising candidate proposed for future investigation.

3.6.1. Internal Cluster Validation Measures

The main motivation for the proposed abrupt change segmentation is to reliably extract stable event-based features suitable for event-clustering and load recognition. This is not limited to the segmented transient phase but also equally apply to steady-state features as the ones in Section 3.5.

In validating the claimed advantage, we conduct the following experiment. With the extracted features[12] defined by Equation (3.9), event-based clustering is performed as detailed in Section 3.6 and cluster validation measures are estimated.

[11]Where M is overloaded at this stage to denote the number of clusters of event-based features rather than the image of a time series sequence on the spatial dimensions.

[12]With the minor difference that cluster cores are considered for *representativeness* of each cluster set rather than all cluster members.

Estimated measures are then compared with those obtained upon replacing the proposed adaptive segmentation with a fixed length τ for the transient phase resulting the simple feature extraction $\mathbf{d}\boldsymbol{x} = \boldsymbol{x}(t+\tau) - \boldsymbol{x}(t)$ for the event detected at time t.

As a measure of extracted feature stability, we leverage *internal* clustering measures. Internal clustering measures primarily quantitatively validate two aspects the estimated clustering structure, namely, intra-cluster compactness and inter-cluster separation[13]. In the sequel, we first introduce the adopted validation measures and then discuss obtained results on the two considered datasets.

Let $\mathcal{X} = \{\boldsymbol{x}^{(n)}\}_{n=1}^{N}$ denote the set of N d-dimensional feature vectors to be clustered. A clustering structure is denoted by $\mathbb{Y} = \{y_m\}_{m=1}^{M}$ which is partitioning $\mathcal{X}$ into M mutually exclusive non-empty subsets[14] denoted by $\mathcal{Y}_m$ for $1 \leqslant m \leqslant M$. Let further $\boldsymbol{c}$ denote the center of the dataset $\mathcal{X}$ estimated as its empirical mean $(1/N)\sum_{\boldsymbol{x}\in\mathcal{X}} \boldsymbol{x}$ and in a like manner $\boldsymbol{c}_m$ for each subset $\mathcal{Y}_m$.

The first measure is the standard R^2 (aka R-squared) statistic commonly adopted in regression tasks. R^2 is defined (similar to one of its definitions in regression) as the ratio between residuals' variance to the variance of the dataset samples

$$\mathrm{R}^2 = 1 - \frac{\mathrm{VAR}_{\text{residuals}}}{\mathrm{VAR}_{\text{data}}} = 1 - \frac{\sum_m \sum_{\boldsymbol{x}\in\mathcal{Y}_m} ||\boldsymbol{x} - \boldsymbol{c}_m||_2^2}{\sum_{\boldsymbol{x}\in\mathcal{X}} ||\boldsymbol{x} - \boldsymbol{c}||_2^2} \quad \in [0,\ 1] \tag{3.12}$$

where $\mathrm{VAR}_{\text{residuals}}$ are defined as the squared deviations of each data point from its cluster mean $\boldsymbol{c}_m$, while $\mathrm{VAR}_{\text{data}}$ is the squared deviation from each data point to the whole data mean $\boldsymbol{c}$. The former is a measure of cluster compactness while the latter is a weak indicator of cluster separation (even though it is not a direct measure of pair-wise cluster separation) [Halkidi et al. 2001, Aggarwal and Reddy 2013]. Here $||\cdot||_2^2$ denotes the squared L_2-norm. The R^2 measure is normalized to the range $[0,\ 1]$ with the optimum value at unity.

Dunn [1974] proposed a validation index that is more explicit about the clustering objectives. Dunn's index $\mathtt{D_I}$ is, therefore, defined as the ratio between inter-cluster separation (measured in the minimum distance between any two distinct clusters

[13]Aggarwal and Reddy [2013] argues that the R^2 measure is an exception that mostly considers cluster compactness. We, however, claim that the R^2 measure indirectly, yet inefficiently, considers cluster separation through the normalizing total sum of squared deviations $\sum_{\boldsymbol{x}\in\mathcal{X}} ||\boldsymbol{x} - \boldsymbol{c}||$.

[14]We ignore unclustered data points (i.e. outliers) in the evaluation procedure.

which is, in turn, defined as the minimum distance between their members) to the intra-cluster compactness (measured as the maximum distance between two members of the same cluster) and is given by

$$\mathtt{D}_{\mathtt{I}} = \frac{\min_m \min_{n \neq m} \min_{\substack{\boldsymbol{x} \in \mathcal{Y}_m \\ \boldsymbol{x}' \in \mathcal{Y}_n}} ||\boldsymbol{x} - \boldsymbol{x}'||}{\max_m \max_{\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{Y}_m} ||\boldsymbol{x} - \boldsymbol{x}'||} \tag{3.13}$$

Intuitively, Dunn's index is a non-negative measure that attains a higher value for a more fitting clustering structure.

In a like manner, Xie and Beni [1991] proposed a cluster validity measure quantifying the ratio between compactness and separation[15]. The former is estimated as the mean of squared deviations of each data point from its cluster representative. Separation, however, is estimated as the minimum distance between any two distinct clusters estimated by the distance between their representative points. Xie-Beni index $\mathtt{XB}_{\mathtt{I}}$ is defined as

$$\mathtt{XB}_{\mathtt{I}} = \frac{\frac{1}{N} \sum_m \sum_{\boldsymbol{x} \in \mathcal{Y}_m} ||\boldsymbol{x} - \boldsymbol{c}_m||_2^2}{\min_m \min_{n \neq m} ||\boldsymbol{c}_n - \boldsymbol{c}_m||_2^2} \tag{3.14}$$

and is similarly non-negatively valued with a lower value indicating a better clustering structure.

The final measure is proposed by Davies and Bouldin [1979] which is a more flexible validation measure that assigns a similarity value to each cluster (rather than a global compactness to separation value) that is again estimated as a compactness to separation ratio. Davis and Bouldin index $\mathtt{DB}_{\mathtt{I}}$ is then estimated as the average of all cluster similarities so that

$$\mathtt{DB}_{\mathtt{I}} = \frac{1}{M} \sum_m \max_{n \neq m} \left(\frac{\left(\frac{1}{|\mathcal{Y}_m|} \sum_{\boldsymbol{x} \in \mathcal{Y}_m} ||\boldsymbol{x} - \boldsymbol{c}_m||_2\right) + \left(\frac{1}{|\mathcal{Y}_n|} \sum_{\boldsymbol{x} \in \mathcal{Y}_n} ||\boldsymbol{x} - \boldsymbol{c}_n||_2\right)}{||\boldsymbol{c}_m - \boldsymbol{c}_n||_2} \right) \tag{3.15}$$

and is likewise non-negatively valued with a lower value indicating a better clustering structure.

[15]Observe the reciprocal relationship to Dunn's index.

Table 3.2.: Internal cluster validation measures on BLUED with a fixed length τ window for transient phase segmentation (first three columns of each phase) against our proposed adaptive segmentation algorithm (right-most column of each phase). The logarithm function $\log(\cdot)$ is taken to base 10.

	Validity Index	$\tau = 0.33$ seconds	$\tau = 0.5$ seconds	$\tau = 1.0$ seconds	$\tau = 1.5$ seconds	**ours**
Phase A	R^2	0.990	0.998	**0.999**	**0.999**	**0.999**
	$\log(\mathtt{D}_\mathtt{I})$	-10.567	-8.100	-6.768	-5.794	**-4.000**
	$\log(\mathtt{XB}_\mathtt{I})$	2.923	0.119	0.314	0.748	**-0.204**
	$\log(\mathtt{DB}_\mathtt{I})$	3.351	2.357	1.643	1.582	**1.317**
Phase B	R^2	0.963	0.975	0.990	0.983	**0.999**
	$\log(\mathtt{D}_\mathtt{I})$	-12.058	-13.716	-9.852	-10.120	**-5.084**
	$\log(\mathtt{XB}_\mathtt{I})$	7.425	6.471	5.971	6.223	**1.360**
	$\log(\mathtt{DB}_\mathtt{I})$	3.465	3.233	2.803	2.750	**1.807**

3.6.2. Empirical Validation

Table 3.2 shows a list of clustering validations performed at various fixed-length transient phase segmentation depicted against our adaptive segmentation algorithm. As clearly observed, our adaptive segmentation algorithm consistently outperforms fixed-length segmentation on all validation measures and for both phases. The advantage is more emphasized in the second phase of the dataset (that is, Phase B) which includes a wider variety of loads with almost twice the number of events increasing the risk of simultaneous events[16]. Further results on BLUED can be found in Appendix B.

Figure 3.5 depicts a similar evaluation framework on the three-phase commercial building dataset. The figure reasserts our previous claim from the BLUED dataset evaluation that adaptive segmentation consistently outperforms fixed-length feature extraction for steady-state features. Such promising results raise even higher expectations upon utilization of transient features but this is outside the scope of this work.

[16]That is, events violating the switch continuity principle by occurring within a very short time interval that are resulting in either missed detections or distorted feature vectors.

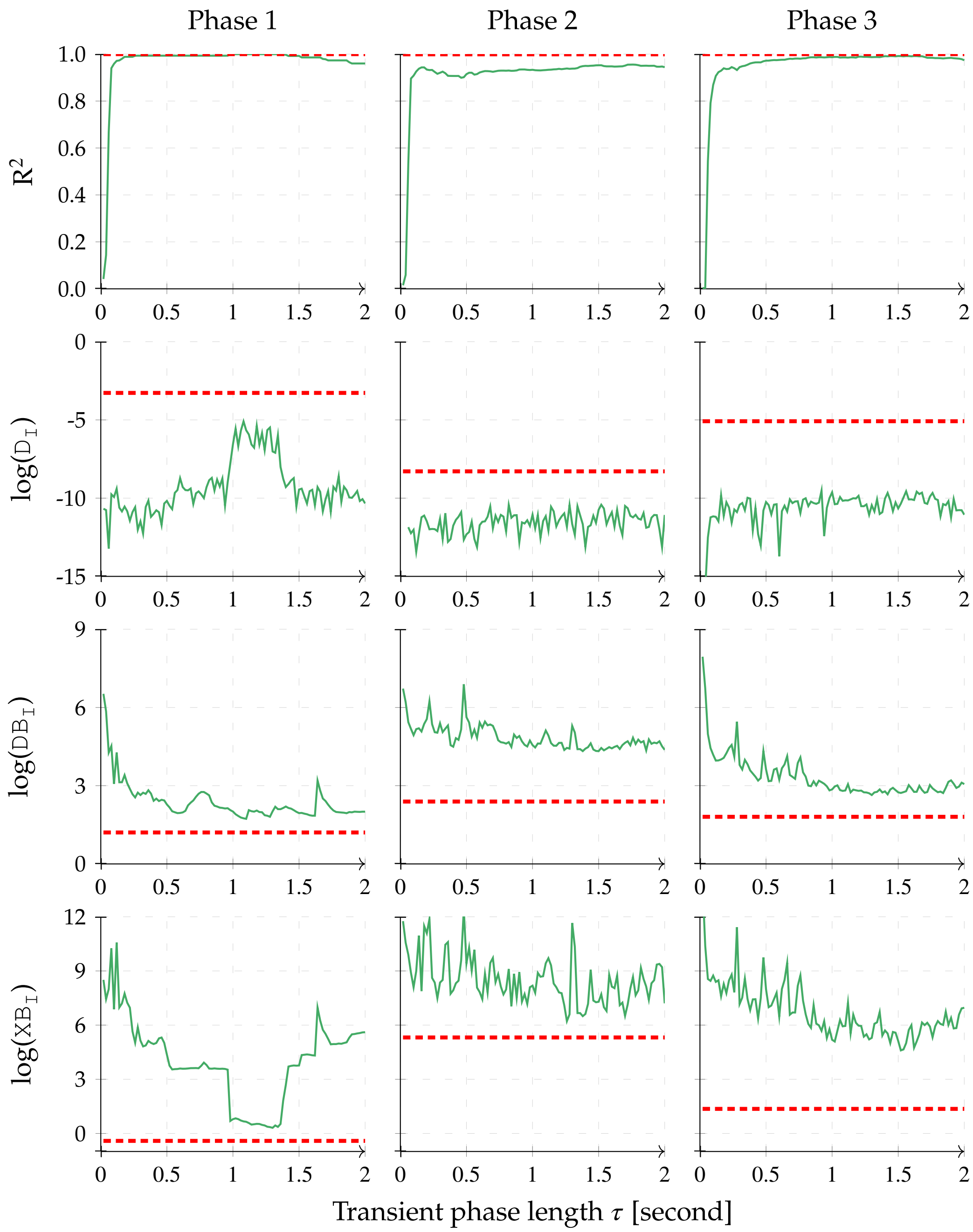


Figure 3.5.: Internal cluster validation measures estimated on the three-phase commercial dataset. Red lines represent the measures of our proposed adaptive segmentation algorithm while green curves are those measures estimated from a fixed window length τ up to 2 seconds.

3.7. Load Recognition

In the event clustering stage, events were split in groups based on the similarity between their extracted feature vectors. The objective at the load recognition stage is to group detected events based on the likelihood that they were generated from the same end-use load. In this section, we propose a load recognition stage based on a set of engineered rules to estimate the number of monitored loads and map events (and event clusters) to inferred loads.

Due to the lack of labeled data, we limit this stage to *on-off* load recognition only. Worth mentioning here is that recognition is only performed to the level of anonymous labeling (that is, unnamed load instances). Identifying human-sensible load identifiers requires labeled data which is assumed to be missing in this unsupervised disaggregation framework.

The proposed load recognition stage leverages both individual event features (referred to hereafter as *event-level* matching and is discussed in detail in Section 3.7.1) and *cluster-level* characteristics (which is detailed in Section 3.7.2) in two matching principles. The first is the ground-state matching principle which is applied to individual events and propagated subsequently across events' clusters. The second principle is based on the zero-sum assumption of multi-state loads and is applied to cluster-level features and exploited afterwards in matching individual events.

In the following, we let a subscript arrow distinguish between switch-*on* and switch-*off* events as in $\boldsymbol{x}_{\uparrow}^{(k)}$ for the switch-*on* event detected at time $t^{(k)}$ and assigned to the cluster $y(\boldsymbol{x}_{\uparrow}^{(k)}) \in \mathbb{Y}$ where $\mathbb{Y} = \{y_m\}_{i=1}^{M}$ and likewise for the switch-*off* event $\boldsymbol{x}_{\downarrow}^{(l)}$. Should a distinction become unnecessary, the subscript is simply discarded as in $\boldsymbol{x}^{(k)}$ denoting the k^{th} detected event. A series of N events is denoted by the chain $\boldsymbol{x}^{(1)} \rightarrow \boldsymbol{x}^{(2)} \rightarrow \cdots \rightarrow \boldsymbol{x}^{(N)}$ where the symbol $\cdots \rightarrow \cdots$ connotes temporal succession of detected events. This notation will be enriched as load recognition rules are introduced in the sequel.

3.7.1. Event-level Matching

Prior to discussing proposed rules, we first introduce the concept of a ground state on which the following rules rely. A ground state is defined as the steady-state where all monitored loads are not in use. Ideally, this is a stationary phase of zero power draw. However, a certain category of loads (known as always-*on* loads) prevents observing the ground state according to the given definition. Therefore, we revise the earlier definition of a ground state to be the state where all event-generating loads are not in use. This excludes always-*on* loads from the proposed load recognition but this is a tolerable omission as shall be discussed in Chapter 5.

A ground state detected after the k^{th} event is denoted by $\boldsymbol{x}^{(k)} \nrightarrow \boldsymbol{x}^{(k+1)}$ with the symbol $\nrightarrow$ representing the detected ground state. Ground states, therefore, split the N-event series into a set of sub-series surrounded by ground states from both endpoints and referred to henceforth as entities. In the earlier example, the ground state detected after the k^{th} event results in two entities. The first is the k-event sub-series $\boldsymbol{x}^{(1)} \rightarrow \cdots \rightarrow \boldsymbol{x}^{(k)}$ and the second is the $N - k$ sub-series $\boldsymbol{x}^{(k+1)} \rightarrow \cdots \rightarrow \boldsymbol{x}^{(N)}$.

Section 3.7.1 depicts our running example (a 5-hour aggregate load profile extracted from the first phase of the BLUED dataset) for illustrating the rule-based load recognition stage. Observable in this illustrative example multiple ground states that segments the entire 5-hour signal into temporally disjoint entities. This leads to the first rule.

Rule 1: *Events from different entities can not be matched.*

Matched events are those attributed to the same duty cycle (that is, a single continuous interval of operation) of a load. The first rule stems directly from the definition of the ground state since no detectable load is permitted to remain operational during a ground state. Hence if a load is switched-*on* during an entity, it must have been switched-*off* prior to the succeeding ground state.

In Section 3.7.1, seven entities can be observed, with the shortest comprising a pair of state-change events, while the longest is composed of 6 events in this illustrative example. Event matching can be simplified to within-entity search according the aforementioned rule.

Rule 2: *The on-off entity identified by the pattern* $\nrightarrow \boldsymbol{x}_{\uparrow}^{(k)} \rightarrow \boldsymbol{x}_{\downarrow}^{(l)} \nrightarrow$ *implies a matching between the* k^{th} *and* l^{th} *events.*

In words, if an entity comprises solely a switch-*on* event followed by a switch-*off* event, then both events are attributed to the same duty cycle of a load. This of

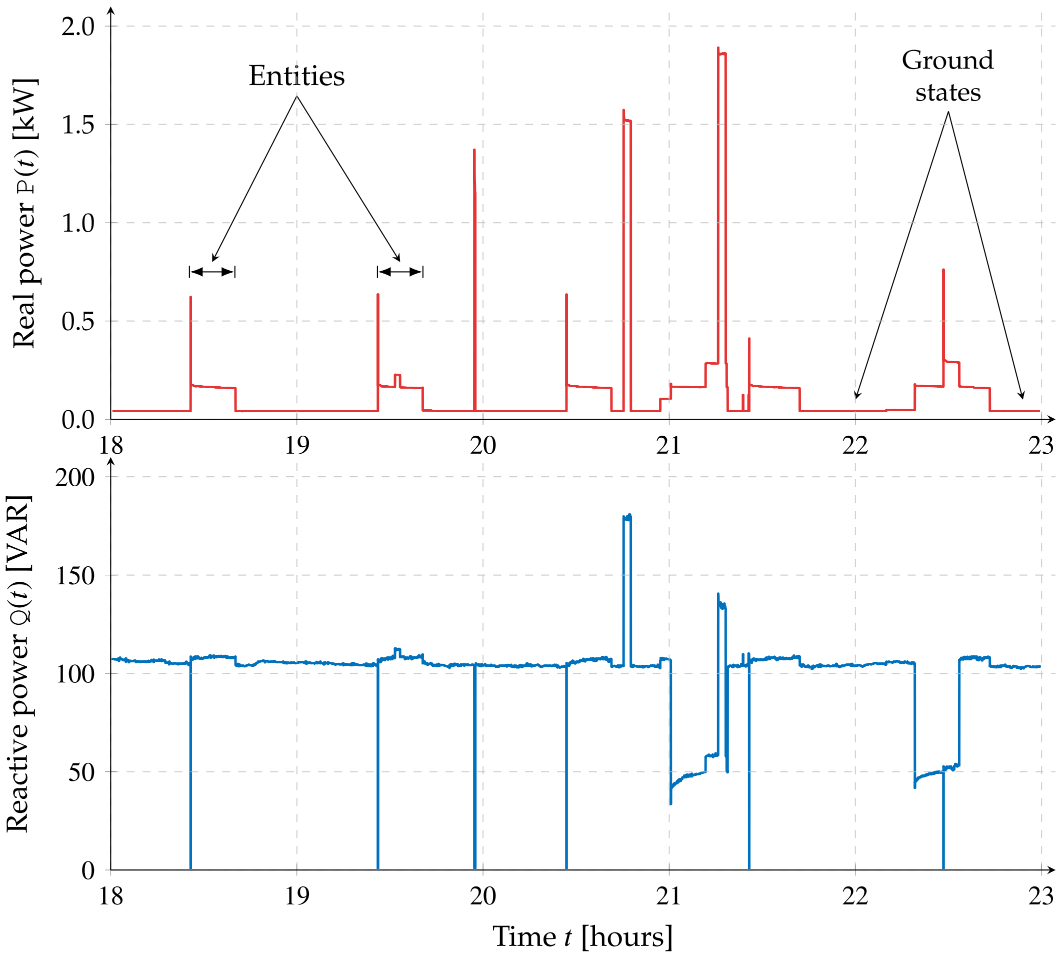


Figure 3.6.: A segment of the aggregate load profile (real power P top, and reactive power Q bottom) from the first phase (phase A) of the BLUED dataset [Anderson et al. 2012]. The segment is extracted from the 18th to the 23rd hour of the dataset, and will be our running example for illustrating various load recognition rules. Ground state power draw for this phase is ~ 40 Watt, and the figure depicts eight instances thereof which, in turn, segments the load profile into seven entities. The figure additionally shows an false detection at time $t \sim 22.17$ hours, and a distorted feature vector due to simultaneous *off* events at time $t \sim 21.31$ hours.

course assumes an ideal event detection[17]. Since an ideal event detector is not guaranteed, we complement the second rule with few auxiliary conditions. The first is

[17]At least with perfect retrieval.

that matched events belong to different clusters[18] (that is $y(\boldsymbol{x}^{(k)}) \neq y(\boldsymbol{x}^{(l)})$), their clusters' cardinalities are adequately matching, and they loosely[19] but adaptively satisfy the zero sum constraint as in $|\texttt{dP}^{(k)} + \texttt{dP}^{(l)}| \leqslant \frac{1}{2}(|\texttt{dP}^{(k)}| + |\texttt{dP}^{(l)}|)$ where $\texttt{dP}^{(k)}$ denotes the change in real power of the kth event[20]. Matched events are represented by $\begin{tikzcd} \boldsymbol{x}^{(k)} \arrow[r] \arrow[rr, dashed, bend left] & \cdots \arrow[r] & \boldsymbol{x}^{(l)} \arrow[ll, dashed, bend left] \end{tikzcd}$ in the event series and denoted in text by $\boldsymbol{x}^{(k)} \leftrightarrow \boldsymbol{x}^{(l)}$.

Section 3.7.1 depicts at least[21] four pairs of matched events via direct application of Rule 2. These are the 1st, 3rd, 4th, and 6th entities.

Rule 3: *If the two events $\boldsymbol{x}^{(k)}$ and $\boldsymbol{x}^{(l)}$ of distinct clusters belong to the same load (that is, matched), then so does their corresponding clusters $y(\boldsymbol{x}^{(k)})$ and $y(\boldsymbol{x}^{(l)})$ should none happen to be the noise set y_0. The two clusters are hence declared as matching.*

Stated differently, if Rule 2 declares a matching between two events indicating that they potentially belong to the same load, then their corresponding clusters also belong to the same load (unless either of the two events is unclustered; that is, belongs the noise set y_0). Equivalently, Rule 3 is expressed as $\boldsymbol{x}^{(k)} \leftrightarrow \boldsymbol{x}^{(l)} \implies y(\boldsymbol{x}^{(k)}) \leftrightarrow y(\boldsymbol{x}^{(l)})$ where $y(\boldsymbol{x}^{(k)}) \neq y(\boldsymbol{x}^{(l)}) \neq y_0$. Practically, we augment Rule 3 with a minimum required number of recurrences per cluster pair in order to be accepted so as to reduces the number of faulty matches. This threshold value is set in our experiments to 3 event matches per cluster pair. Rule 2 and 3 imply jointly that two matching clusters $y_m \leftrightarrow y_n$ must be a pair of *on-off* clusters (that is, one is a cluster of *on*-events and the other is of *off*-events). Similar to events denotation, subscript arrows will distinguish between *on* and *off* clusters, as in $y_{m,\uparrow}$ and $y_{n,\downarrow}$, respectively.

[18]That includes the case where both belong to the noise cluster since matching these two events is of a very limited value.

[19]We use this term to indicate that the sum of power changes can deviate from zero in a relatively wide range and that we only impose this assumption on the change in real power since reactive power can deviate largely as in loads with power storage capabilities for instance.

[20]A zero-sum constraint is originally stated as $|\texttt{dP}^{(k)} + \texttt{dP}^{(l)}| \to 0$ and a relaxed version that is normally adopted in practice is simply comparing the absolute sum or the power changes against a threshold ϵ as in $|\texttt{dP}^{(k)} + \texttt{dP}^{(l)}| \leqslant \epsilon$. To make this matching adaptive to power change, we choose a threshold ϵ that is a function of the absolute changes such that $\epsilon = \frac{1}{\theta}(|\texttt{dP}^{(k)}| + |\texttt{dP}^{(l)}|)$. This concept applies also to similar matching rules as, for example, in Equations (3.16) and (3.17).

[21]As shall be illustrated in the load recognition algorithm, introduced rules will be applied recursively after discarding matched events from the sequence of detections. This leads to new cases where matching rules can be further observed. For instance, recursively applying Rule 1 and Rule 2 leads to 10 cases of matched event pairs, and leaves out two unmatched events, a simultaneous event, and a false detection.

Since this work is limited to *on-off* loads, we only permit unique pairs of cluster matches. Stated more explicitly, if the m^{th} and n^{th} clusters are matched $y_m \leftrightarrow y_n$ where $n, m > 0$ then neither is permitted to match any other cluster (that is, $y_m \nleftrightarrow y_k$ for $k \neq n$ and likewise for y_n). Should a conflict arise e.g. due to missed detections, the cluster pair with the maximum number of matching recurrences is selected.

This rule represents the first step in promoting an event-level matching to cluster-level. The following two rules, then, governs how this cluster-level matching is spread across cluster members.

Rule 4: *For any cluster matching $y_{m,\uparrow} \leftrightarrow y_{n,\downarrow}$, an event $\boldsymbol{x}_{\uparrow}^{(k)}$ of the on-cluster $y_{m,\uparrow}$ forward matches the nearest succeeding event belonging to the off-cluster $y_{n,\downarrow}$. Similarly, an event $\boldsymbol{x}_{\downarrow}^{(l)}$ of the off-cluster $y_{n,\downarrow}$ backward matches the nearest preceding event belonging to the on-cluster $y_{m,\uparrow}$.*

Rule 5: *Two events match $\boldsymbol{x}_{\uparrow}^{(k)} \leftrightarrow \boldsymbol{x}_{\downarrow}^{(l)}$ if $\boldsymbol{x}_{\uparrow}^{(k)}$ forward matches $\boldsymbol{x}_{\downarrow}^{(l)}$ and $\boldsymbol{x}_{\downarrow}^{(l)}$ backward matches $\boldsymbol{x}_{\uparrow}^{(k)}$ backward.*

To illustrate Rules 4 and 5 we use the following simplified example. Assume a cluster match $y_{1,\uparrow} \leftrightarrow y_{2,\downarrow}$ and let a 4-event series comprise two events of the *on*-cluster followed by two events of the *off*-cluster so that the series is expressed as $\boldsymbol{x}_{\uparrow}^{(1)} \rightarrow \boldsymbol{x}_{\uparrow}^{(2)} \rightarrow \boldsymbol{x}_{\downarrow}^{(3)} \rightarrow \boldsymbol{x}_{\downarrow}^{(4)}$. Denote a forward match by $\begin{tikzcd} \boldsymbol{x}^{(k)} \arrow[r] \arrow[rr, dashed, bend left] & \cdots \arrow[r] & \boldsymbol{x}^{(l)} \end{tikzcd}$ and a backward match by $\begin{tikzcd} \boldsymbol{x}^{(k)} \arrow[r] & \cdots \arrow[r] & \boldsymbol{x}^{(l)} \arrow[ll, dashed, bend left] \end{tikzcd}$. Rule 4 then states that the series is expressed as

$$\begin{tikzcd} \boldsymbol{x}_{\uparrow}^{(1)} \arrow[r] \arrow[rr, dashed, bend left] & \boldsymbol{x}_{\uparrow}^{(2)} \arrow[r] \arrow[r, dashed, bend left] & \boldsymbol{x}_{\downarrow}^{(3)} \arrow[r] \arrow[l, dashed, bend left] & \boldsymbol{x}_{\downarrow}^{(4)} \arrow[ll, dashed, bend left] \end{tikzcd}$$

and Rule 5 states that only the $\boldsymbol{x}_{\uparrow}^{(2)} \leftrightarrow \boldsymbol{x}_{\downarrow}^{(3)}$ matching is accepted. Worth mentioning is that matches remain only possible within the same entity as stated by the first rule. Admittedly, these two rules represent two main limitations in our unsupervised load recognition stage and an opportunity for future research. Rules 4 and 5 represent a conservative choice that favors the hypothesis of faulty event detection over the claim of load aliasing (that is, multiple loads with similar signatures).

The ground-state event matching procedure is then stated as follows:

1. Given an event series, detect ground states and split entities

2. Apply Rule 2 to each entity to extract event matches

3. Apply Rule 3 to promote event matches to cluster-level

4. Apply Rules 4 and 5 to spread cluster-matches to events
5. Remove matched (and partially matched) events from each entity
6. Repeat from Step 2 until no further matches are possible

and all steps are performed with Rule 1 in mind. Partially matched events are those matched only forwards or only backwards (as in $\boldsymbol{x}_{\uparrow}^{(1)}$ and $\boldsymbol{x}_{\downarrow}^{(4)}$ in the preceding example). Clearly this procedure is overly conservative and extensions can be proposed but are not investigated further in this work and are rather recommended for future investigation. However, and due to the limitations of this matching approach, we additionally complement this approach with the commonly adopted zero-sum principle [Hart 1992] which is introduced in the following.

3.7.2. Cluster-level Matching

The event-level matching discussed in the preceding subsection faces two main difficulties. First it relies on estimating a ground state which require a long monitoring period to be observed. Second, in each loop iteration the matching algorithm begins with a search for two-event entities to apply the event-level matching. Even if a ground state is observed, finding a two-state entity is an assumption not guaranteed to hold. Therefore, once the event-level matching procedure has retrieved all possible matching pairs, further matching possibilities are retrieved using cluster-level features and these are discussed in the following.

In the following, we often distinctly differentiate between *on* and *off* clusters which is intentional since we primarily consider *on-off* loads and hence only permit a matching between a cluster pair with opposing change in power.

Zero sum constraint: Hart [1992] introduced the concept of zero-sum of a duty cycle of a load and this concept has been utilized by many energy disaggregation approaches afterwards (see e.g. [Giri and Bergés 2015, Krall et al. 2016, Azaza and Wallin 2017, Liu et al. 2017]). The concept states that sum of state transitions of a complete duty cycle of an FSM load is nearly zero-valued (assuming nearly constant power draws of an FSM load in each state). While we do not strictly enforce the zero-sum constraint on each event, we adopt a relaxed variant thereof applied to the cluster representatives.

Since we only consider *on-off* loads, a matched pair of clusters are expected to nearly match in their absolute power change $\mathtt{dP}$ and $\mathtt{dQ}$ features. Letting $\mathtt{dP}_{m,\uparrow}$ and $\mathtt{dP}_{n,\downarrow}$ denote the power change features of the $y_{m,\uparrow}$ *on*-cluster and the $y_{n,\downarrow}$ *off*-cluster, respectively, then the condition

$$\left|\,\mathtt{dP}_{m,\uparrow}+\mathtt{dP}_{n,\downarrow}\right| \;\leqslant\; \frac{1}{\vartheta_{\mathtt{dP}}}\Big(\left|\mathtt{dP}_{m,\uparrow}\right|+\left|\mathtt{dP}_{n,\downarrow}\right|\Big) \tag{3.16}$$

(where $\vartheta_{\mathtt{dP}} > 1$) retrieves a list of potential candidates for each cluster $y_{m\uparrow}$ ordered by the sum of power change $|\mathtt{dP}_{m,\uparrow}+\mathtt{dP}_{n,\downarrow}|$. Here $\mathtt{dP}_m$ of the mth cluster is estimated from the cluster mean given by $\boldsymbol{x}_m = 1/|\mathcal{Y}_m| \sum_{\boldsymbol{x}\in\mathcal{Y}_m} \boldsymbol{x}$ where $\boldsymbol{x}_m = [\mathtt{dP}_m, \mathtt{dQ}_m]^{\mathsf{T}}$. A similar list of potential matching candidates are retrieved using a constraint on the change in reactive power given analogously by

$$\left|\,\mathtt{dQ}_{m,\uparrow}+\mathtt{dQ}_{n,\downarrow}\right| \;\leqslant\; \frac{1}{\vartheta_{\mathtt{dQ}}}\Big(\left|\mathtt{dQ}_{m,\uparrow}\right|+\left|\mathtt{dQ}_{n,\downarrow}\right|\Big) \tag{3.17}$$

(where $\vartheta_{\mathtt{dQ}} > 1$) and similarly ordered by the sum of change $|\mathtt{dQ}_{m,\uparrow}+\mathtt{dQ}_{n,\downarrow}|$. Conditions 3.16 and 3.17 represent an adaptive, weak version of the zero-sum constraint as they state that power changes in the *on*- and *off* clusters should be close but not necessarily zero-valued. Closeness here is stated as that one event should be in the range of $(\vartheta_{\mathtt{dP}}-1)/(\vartheta_{\mathtt{dP}}+1)$ to $(\vartheta_{\mathtt{dP}}+1)/(\vartheta_{\mathtt{dP}}-1)$ times the other for them to qualify for potential matching (and likewise for the reactive power). In all our experiments, both parameters are identically set to $\vartheta_{\mathtt{dP}} = \vartheta_{\mathtt{dQ}} = 3$.

Potential event matching: A second condition for a cluster-level matching is the anticipated event pairs that could be matched according to Rules 1, 4, and 5. In other words, if the two *on*- and *off*-clusters indeed correspond to the same load, it is likely that their members will be observed in ordered *on-off* pairs and within the same entity of operation. Therefore, the condition

$$\sum_{\substack{\boldsymbol{x}_\uparrow\in\mathcal{Y}_m\\ \boldsymbol{x}_\downarrow\in\mathcal{Y}_n}} \mathbb{1}\big(\boldsymbol{x}_\uparrow \leftrightarrow \boldsymbol{x}_\downarrow\big) \;\leqslant\; \vartheta_e\cdot\max\big(|\mathcal{Y}_m|,\,|\mathcal{Y}_n|\big) \tag{3.18}$$

(where $0 < \vartheta_e \leqslant 1$ and $\mathbb{1}(\boldsymbol{x}_\uparrow \leftarrow \boldsymbol{x}_\downarrow)$ indicates the number of possible matches) similarly retrieves for a given cluster y_m a list of potential clusters for matching ordered by the ratio of anticipated event matches to the cluster cardinality. This threshold is set in our experiments to $\vartheta_e = 40\%$.

The cluster-level matching procedure is then detailed as follows:

1. Set tolerance level $l = 1$
2. For each unmatched *on*-cluster $y_{m,\uparrow}$
3. Retrieve three ordered lists of matched *off*-clusters based on the Conditions 3.16, 3.17, and 3.18
4. If the l^{th} first elements of each list has a candidate in common (e.g. $y_{n,\downarrow}$), then declare the matching $y_{m,\uparrow} \leftrightarrow y_{n,\downarrow}$.
5. Repeat from Step 2 unit no further matching is possible.
6. Increment the tolerance level $l \leftarrow l + 1$
7. Repeat from Step 2 until a maximum tolerance (i.e. for $l \leqslant L$)

At the end of this stage, each pair of matched clusters (either indirectly from the preceding event-level stage or the current cluster-level matching procedure) are declared as a distinct load for which a power trace (that is, load profile or the time series draw of the real power) is to be reconstructed.

We defer evaluation of this and the following stage (that is, load profile reconstruction) to the end-to-end validation of the entire disaggregation pipeline presented in Section 3.9.

3.8. Load Profile Reconstruction

For each detected *on-off* load, the load profile is reconstructed with the assumption of a linear power draw from each *on*-event to its matched *off*-event. Additionally, a power surge (a spike of the power draw during the *on*-transient phase) extracted from the *on*-event transient segmentation is injected into the reconstructed time series. Load profiles are reconstructed at a frequency of 1 Hz.

3.9. Experiments and Results

In this section we provide qualitative evaluation of the proposed end-to-end disaggregation framework applied to the commercial dataset. Quantitative evaluations are outside the scope of this work, and are delegated to a solid future work on energy disaggregation validity measures which remains, to our knowledge, a debate within the energy disaggregation to date.

Figure 3.7 shows the aggregate real power signal of the commercial data on a 6-day period in January 2015. The proposed unsupervised energy disaggregation frameworks reaches up to 20 detected loads of which we provide a single example depicted in Figure 3.8 and further results can be found in Appendix C.

Figure 3.8 shows the disaggregation of a load commonly used during the evening on workdays. Amongst the list of sub-metered loads in that dataset, the best observed candidate is the dishwasher whose sub-metered profile is shown in Figure 3.9. As observable from the figure, our framework was able to precisely estimate the start-time of each usage cycle and, to a large extent, the whole duration of operation. Observable yet is that the reconstructed profile is much smoother than the real one (with many power surges missed) in addition to missing a second state of this multi-state load. Please note the difference in power draw levels is mostly attributed to calibration artifacts since different power meters have been used for the aggregate and sub-metered signals.

3.10. Conclusion

In this chapter, we proposed a multi-stage unsupervised energy disaggregation framework which comprised five stages, namely, a clustering-based event detection stage, feature extraction, non-parametric event-based clustering stage, event-matching for *on-off* load recognition, and finally inverse load profile reconstruction.

Being unsupervised, the introduced framework relied in general on repeated pattern detection and matching. However, it additionally leveraged domain knowledge encoded in a set of rules or heuristics that were proven to be valuable in ad-

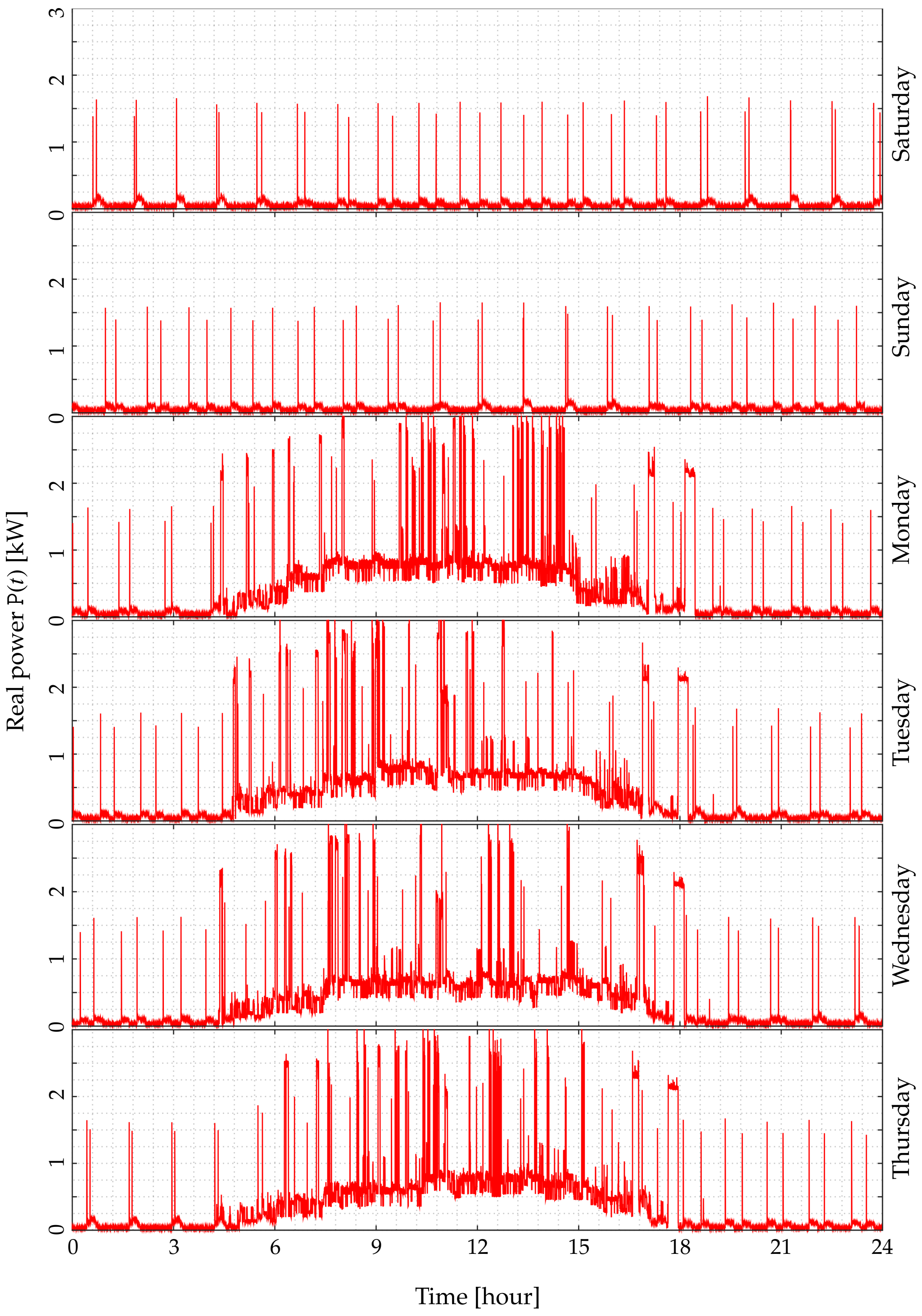


Figure 3.7.: Aggregate real power of the 1st phase of a commercial building.

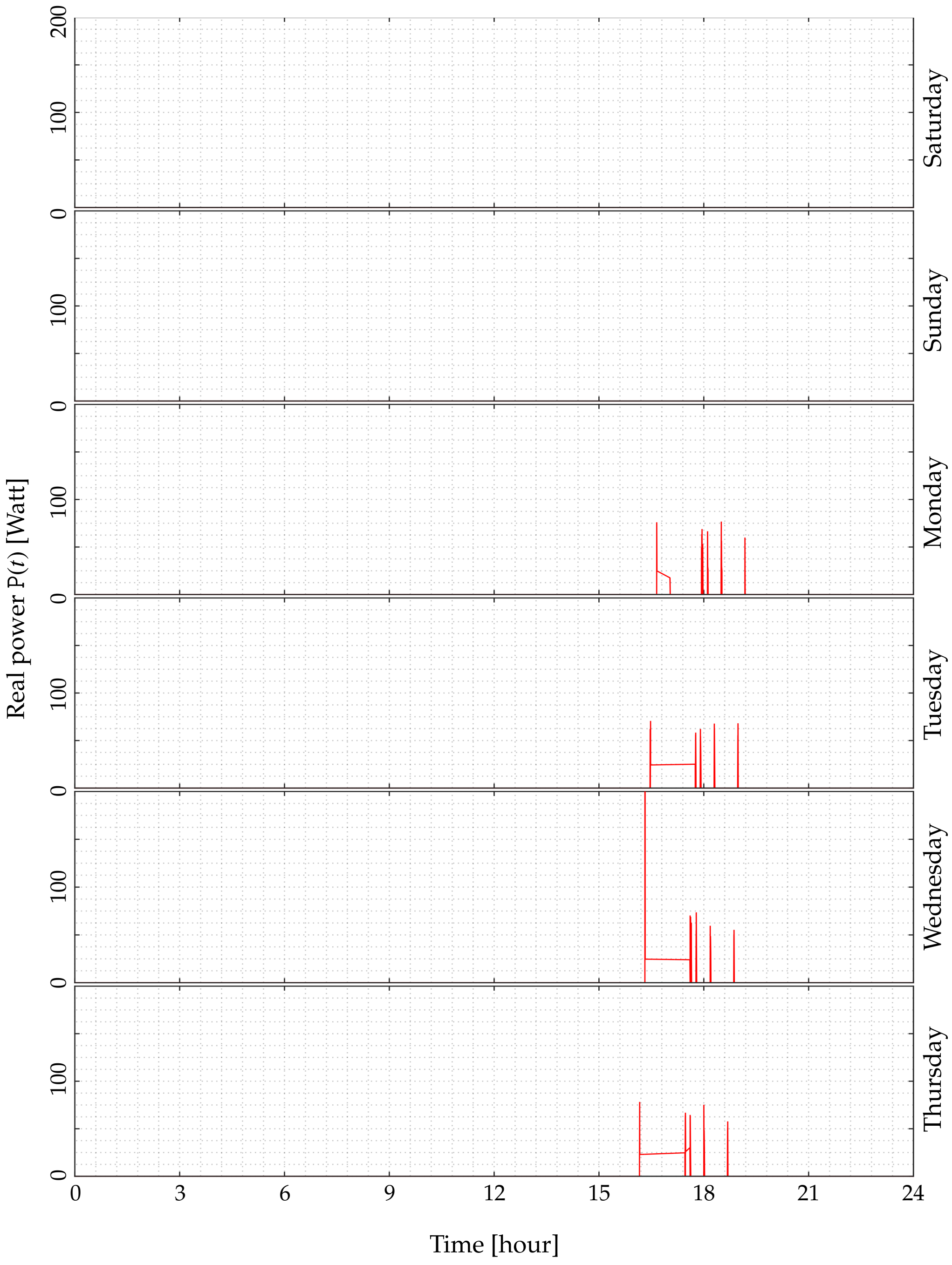


Figure 3.8.: Reconstructed profile of a detected load in the 1st phase of the commercial data with estimated energy 0.1 kW·H, 6-day usage time ∼ 4.6 hours (that is around 3.2% of the disaggregation period), and around 4.1% of detected events.

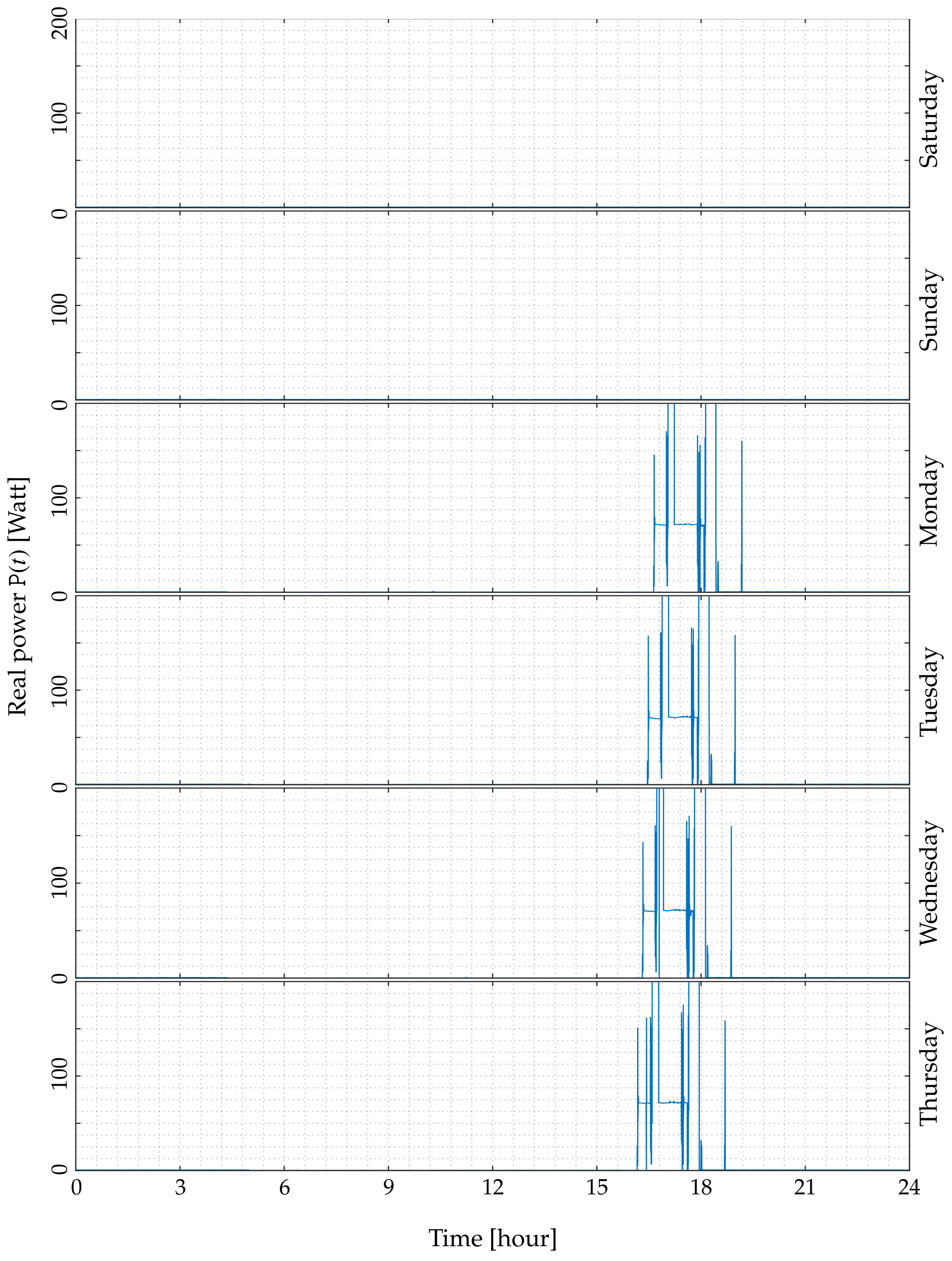


Figure 3.9.: Best-match sub-metered load profile to the detected load in Figure 3.8. Meter calibration artifacts results in the observed scale difference between the aggregate data meter and the individual load sensors.

dressing the load disaggregation task. In spite of the introduced rules and heuristics, the algorithm featured a high level of interpretability that, in turn, allowed for ease of hyper-parameter adjustment. Notable yet is the high level of robustness against parameter variations indicating a stable disaggregation performance that rendered this framework equally applicable to both the residential and commercial settings on which the proposed model was indeed empirically validated.

For future work on such a multi-stage framework, probabilistic modeling represents one of the most promising improvements. As a main advantage, a probabilistic framework permits the system user or domain expert to encode their uncertainty in the system's parameters. Such a probabilistic framework would then reflect user's uncertainty in design parameters to model's uncertainty in predicted outputs which would further promote the level explainability and transparency of the proposed framework. A second advantage of a probabilistic framework is that it permits a propagation of uncertainties from one stage to the next so as to e.g. recall missed detections at a higher level of abstraction.

Extending the proposed approach to multi-state and variable loads is another possible direction. Yet, the work introduced in this chapter reflects on some of challenges in unsupervised energy disaggregation. Therefore, we proceed in the following chapters with gradual increase of labeled data for energy disaggregation from event-based labels in Chapter 4 to sub-metered loads in Chapter 5.

Chapter 4.

Semi-Supervised Energy Disaggregation

Ubiquity of unlabeled measurements, along with scarcity of labeled data, motivated a machine learning paradigm known as *semi-supervised learning* (SSL). SSL leverages the ubiquity of unlabeled data in guiding the model-under-training towards more reliable generalization performance especially when labeled data is not adequately available. Indeed, some assumptions have to hold for this learning paradigm to achieve its targeted performance gain. However, these assumptions are neither restrictive nor unreasonable in many practical scenarios. In general, it is assumed that structure or dynamics of the data in the feature space is relevant to the structure of the label space. In simpler terms, it is assumed that signatures or features having the same label tend to cluster together in the feature space. In that case, revealing the feature space structure to an appropriate learning model assists in inferring the structure of the label space.

This chapter presents our work in the direction of semi-supervised energy disaggregation [Barsim and Yang 2015]. The proposed learning paradigm reduces labeling and training efforts of standard supervised learning models, mitigates the problem of labeled data scarcity, and enables a form of online learning as more unlabeled data become available post-deployment. Experimental validation shows that SSL can achieve a notable boost in performance in situations of severe scarcity of labeled data.

This chapter is organized as follows. Section 4.1 gives a brief overview of the SSL and an example of this learning setting, namely, self-training. Section 4.2 reviews prior art on energy disaggregation that adopted SSL. In Section 4.4, our proposed

SSL model for energy disaggregation is detailed. Experimental validation on a real-world household dataset, namely, the BLUED dataset [Anderson et al. 2012] is provided in Section 4.7. Finally, Section 4.8 concludes this chapter.

This work was published by Barsim and Yang [2015], and formulations, phrasing, and results are partially adopted from our publication.

4.1. Introduction

Semi-supervised learning (SSL) refers to a set of machine learning paradigms that either assist an unsupervised component with externally labeled data or accommodate a supervised learner with unlabeled observations [Chapelle et al. 2006, Goldberg 2010]. SSL attempts to infer valuable information about the structure of the label space through exploring the feature space, assisted by constraints on the former. Such constraints are provided either explicitly or through scarcely labeled data. The main objective of SSL tools is to address the scarcity of labeled data, and this is achieved by leveraging the ubiquity of unlabeled observations to probe and explore the feature space for valuable structures.

Formally stated, SSL attempts to estimate the conditional distribution $p(y \mid x)$ of the labels y given observations x through samples obtained from the joint distribution $p(x, y)$ in addition to a ubiquity of samples from the marginal distribution $p(x)$ [Schoelkopf et al. 2012]. We refer to samples acquired from the marginal $p(x)$ as unlabeled observations while labeled data is the sample set obtained from the joint $p(x, y)$.

In the last three decades, SSL algorithms have been utilized in various applications with different models including self-training [Yarowsky 1995b], co-training [Blum and Mitchell 1998], semi-supervised support vector machines [Joachims 1999], and generative mixture models [Nigam et al. 2000]. We refer the interested reader to [Chapelle et al. 2006, Goldberg 2010] for a more comprehensive review on SSL.

Plentiful examples can be quoted where unlabeled observations overwhelm labeled data for a machine learning problem. Of a particular relevance to our discussion is the case of energy disaggregation. In energy disaggregation, labeled data is either segregated signals of sub-metered loads for eventless disaggregation or manually labeled events for event-based energy disaggregation systems. Respectively, unlabeled data is either aggregate signals or their detected events. Undoubtedly, acquiring segregate load profiles is considerably more invasive, laborious, and time-consuming than recording aggregate signals which is a by-product of the smart meter deployment. Similarly, detecting abrupt changes in an aggregate signal is much cheaper than manually associating each detection to the causing load.

Self-training [Scudder 1965, Fralick 1967, Agrawala 1970, Amini and Gallinari 2002] is one of the simplest SSL algorithms in which a probabilistic model is initially trained on the limited amount of labeled data, and then iteratively re-trained and updated based on both the externally labeled data, and its own high-confidence predictions. A probabilistic model is a model that quantifies its confidence (or equivalently uncertainty) in its own prediction. The self-training SSL paradigm has been adopted in a number of machine learning problems such as named entity classification [Collins and Singer 1999], object detection [Rosenberg et al. 2005], and computational linguistics [Yarowsky 1995a].

In this work, we prove the effectiveness of SSL for event-based energy disaggregation. To this end, we propose a self-training model for event-based classification and empirically validate this model on a publicly available energy dataset. The proposed model consists of a support vector machine (SVM) along with a nearest neighbor rule for label propagation as shall be detailed shortly.

4.2. Literature Review

It is important to remark the terminology peculiars resulting from redefining some machine learning concepts for energy disaggregation. As introduced in the preceding section, we use the term *"semi-supervised"* as defined in the domain of machine learning in general, namely, learning from labeled and unlabeled data. Works on energy disaggregation, however, tend often to use the same term to indicate reduced level of intrusiveness upon deployment in a target building (see Section 1.2.2 for a terminology discussion). In this section, we limit our review to prior art on semi-supervised disaggregation methods as defined from a machine learning perspective.

Our work [Barsim and Yang 2015] along with the independent and shortly preceding works by Li et al. [2015] and Iwayemi and Zhou [2017][1] were, to our knowledge, the first of their kind in proposing SSL for energy monitoring and disaggregation. All three works utilized the self-training paradigm but varied in their base learner. An expectation maximization (EM) classification model is proposed by Li et al. [2015], a *k*-nearest neighbor *k*NN by Iwayemi and Zhou [2017], and an SVM in our earlier work [Barsim and Yang 2015]. Iwayemi and Zhou [2017] additionally complement our work through an elaborate search for an efficient stopping criterion. Stopping in their work was considered an anomaly detection problem where a Gaussian model is fit to labeled and augmented samples. Using the fitted model, unlikely data points are prevented from augmenting the labeled set[2]. The framework is evaluated on the REDD dataset [Kolter and Johnson 2011] in a transductive setting (introduced in Section 4.7), and claim minimal degradation in performance (compared to fully supervised models) despite the severe scarcity of labeled data).

Co-Training, a paradigm of SSL [Blum and Mitchell 1998], was introduced by Gillis and Morsi [2016; 2017] for the problem of energy disaggregation. The authors leveraged diversity in class predictions between two different classification models (a decision tree and a *k*NN in this case) trained on distinct subsets of the labeled data [Xu et al. 2012]. The training set for each model is augmented with samples from the unlabeled data for which both learners were in agreement with respect to their class predictions. Re-training then takes place for both models with the newly augmented training set, and the process repeats until a predefined stopping condition (e.g. no further prediction agreements are possible). While experimental validation was limited to synthetic data and a relatively small number of loads, results show comparable performance to fully supervised models.

Fatouh and Nasr [2018] proposed a combination between self-training with decision trees (DT) as base classifiers and active learning. In their approach, the SSL model (in the training phase) is additionally permitted to query an oracle about its least confident predictions. The model is validated on the BLUED dataset [Anderson et al. 2012], and illustrated consistent improvements. Nevertheless, such a setting is more precisely referred to as active learning [Settles 2012].

[1]The work is actually published in December 2015.

[2]Upon detailing the proposed SSL algorithm, it will become evident that self-training is vulnerable to error (that is, false predictions) propagation.

Humala et al. [2018] introduced a model-tuning stage that leverages unlabeled measurements in target buildings for fine-tuning generic load models. After careful selection of temporal sub-segments, a set of parameters are estimated for each load model. A parameter clustering stage takes place afterwards for detection of anomalies and more robust parameterization. Model-tuning is performed upon installation in a new target building or if new loads are introduced into the building without the need for user intervention or intrusive calibration procedures.

4.3. Problem Statement

Since labeled data is expensive to acquire and is oftentimes scarce, we try in this work to answer the following question. *Can a learning model benefit from an abundance of unlabeled data from the target building to recover to the fully supervised performance ?*

When the answer to the first question is positive, we further ask the following question. *Can this model generalize to entirely novel queries without the need for retraining ?* In other words, how much of unlabeled data has to be available during training for the model to generalize well to new, previously unseen data ?

To this end, we limit this study to event-based energy disaggregation with the example of self-training as the chosen SSL paradigm.

4.4. Model Architecture

The proposed model is a self-training wrapper around a maximum margin classifier, namely an SVM, applied on extracted features of abrupt changes (aka events) from the aggregate signals. Figure 4.1 shows a block diagram illustrating the training procedure which will be detailed in the following.

Let $\mathscr{D} = \{(\boldsymbol{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ be a dataset of N samples sampled from the joint distribution $p(\boldsymbol{x}, y)$ such that the nth feature vector $\boldsymbol{x}^{(n)} \in \mathbb{X}$ is associated with the corresponding discrete-valued class identifier (i.e. label) $y^{(n)} \in \mathbb{Y}$. Here we use $\mathbb{X}$ and $\mathbb{Y}$ to denote the input and output spaces, respectively, with the latter being a finite set of M class labels given by $\mathbb{Y} = \{y_m\}_{m=1}^{M}$. Oftentimes, we drop the sample index denotation $\cdot^{(n)}$ for notational convenience. We assume that the first $N_{\mathscr{L}}$ data samples of $\mathscr{D}$

are accessible to the model under consideration while the remaining $N_{\mathscr{U}} = N - N_{\mathscr{L}}$ are query samples whose labels $y^{(n)}$ are to be predicted by the proposed model. Important yet is that the feature vectors $\boldsymbol{x}^{(n)}$ of the query samples are available to the model even during the training phase[3].

In other words, we assume that the dataset $\mathscr{D}$ is split into a labeled subset $\mathscr{L}$ given, for example, by the first $N_{\mathscr{L}} < N$ samples of $\mathscr{D}$ such that

$$\mathscr{L} = \left\{ (\boldsymbol{x}^{(n)}, y^{(n)}) \right\}_{n=1}^{N_{\mathscr{L}}} \tag{4.1}$$

and an unlabeled set $\mathscr{U}$ comprised of only the feature vectors of the last $N_{\mathscr{U}}$ items of the dataset $\mathscr{D}$ such that

$$\mathscr{U} = \left\{ \boldsymbol{x}^{(n)} \right\}_{n=N_{\mathscr{L}}+1}^{N} \tag{4.2}$$

where $N_{\mathscr{L}} + N_{\mathscr{U}} = N$. The labels of the last $N_{\mathscr{U}}$ are discarded and may be used merely for evaluation purposes[4]. We additionally refer to $\mathscr{L}$ as the seed labeled set, and denote it by $\hat{\mathscr{L}}_0 = \mathscr{L}$ for the iterative training procedure detailed later.

We limit this study to non-abstaining classification models[5]. As a result, we state our first requirement on the seed labeled set as follows

$$\mathbb{Y} = \{y_m\}_{m=1}^{M} \equiv \left\{ y^{(n)} \right\}_{n=1}^{N_{\mathscr{L}}} \tag{4.3}$$

that is, class labels in the seed labeled set $\mathscr{L}$ are expected to span the whole label space. Stated explicitly, the seed labeled set is expected to contain at least a single sample from each target class. In the field of energy disaggregation, this constraint assumes either prior knowledge of all target loads or external labeling of newly introduced loads, which is admittedly a limitation of the adopted SSL paradigm along with the non-abstaining SVM as a base model. However, this can be easily

[3]And if the model is merely evaluated based on these query samples, then this is a setup referred to as transductive semi-supervised learning (see Section 4.7.1)

[4]In practice, these labels will not be available to the oracle in the first place and are to be estimated by the trained model. We emulate this scenario by ignoring some labels of an entirely labeled dataset $\mathscr{D}$.

[5]Abstaining classification models are those models that support *out-of-distribution* detection. Loosely speaking, these models are allowed to *reject* prediction if the query sample is most likely to be an outlier, or an observation whose label has not been included in the training set.

addressed if an abstaining classification model (for example, a Bayesian classifier[6] or a DT[7]) are utilized instead.

Let $\mathfrak{h} : \hat{\mathscr{L}} \mapsto \hat{g} = \mathfrak{h}[\hat{\mathscr{L}}]$ denote a fully supervised training algorithm that estimates a classification function $\hat{g} : \mathbb{X} \to \mathbb{Y}$ from a set $\hat{\mathscr{L}} = \{(\boldsymbol{x}^{(n)}, y^{(n)})\}_{n=1}^{N_{\hat{\mathscr{L}}}}$ of observation-label pairs. For example, in our experiments each prediction function $\hat{g}$ is a multi-class[8] SVM and $\mathfrak{h}[\hat{\mathscr{L}}]$ is the estimation of their support vectors from the labeled set $\hat{\mathscr{L}}$.

The self-training algorithm then iteratively proceeds for a predefined number of iterations K as follows. At the k^{th} iteration, a fully supervised classification model is trained, class labels of unlabeled samples are predicted, an appropriate subset of these samples is selected, and finally the labeled set is augmented with this selection for the next iteration. These steps shall be formalized in the following.

Initialization: Let the seed set of labeled data $\hat{\mathscr{L}}_0$ comprise the manually labeled samples from $\mathscr{D}$ such that

$$\hat{\mathscr{L}}_0 \equiv \mathscr{L} \tag{4.4}$$

Training: Estimate the prediction function $\hat{g}^{(k)}$ from the labeled set $\hat{\mathscr{L}}_k$

$$\hat{g}^{(k)} = \mathfrak{h}\left[\hat{\mathscr{L}}_k\right] \tag{4.5}$$

for each iteration k where $\hat{\mathscr{L}}_k$ denotes the potentially augmented labeled set at the k^{th} iteration.

Prediction: Estimate class labels of the unlabeled set $\mathscr{U}$ using the estimated prediction function $\hat{g}^{(k)}$

$$\hat{y} = \hat{g}^{(k)}(\boldsymbol{x}) \quad \text{for each } \ \boldsymbol{x} \in \mathscr{U} \tag{4.6}$$

where $\hat{y}$ is the estimated label for the sample $\boldsymbol{x}$ at the current iteration. Note that label prediction $\hat{y}$ of the data sample $\boldsymbol{x}$ can change from one iteration to the next as the prediction function $\hat{g}^{(k)}$ is updated.

[6]Bayesian classifiers are uncertainty-aware models that can quantitatively demonstrate their confidence in their own predictions. This is normally estimable from the model's predictive variance. High predictive uncertainty can be interpreted as an abstaining behavior.

[7]Decision trees can easily abstain if designed specifically for such purposes (e.g. if $x_i < 50$, return "no label").

[8]We use the term *multi-class SVM* to refer to a simplification of the multi-class problem into a set of binary classifications and utilizing binary SVMs thereon [Allwein et al. 2001]. An example was given in Chapter 2 for a one-vs-one construction for binary neural nets.

Selection: Based on a pre-designed selection criterion, select a set of unlabeled samples along with their predicted labels resulting in an augmentation set denoted by $\mathrm{d}\hat{\mathscr{L}}_k$ for the kth iteration. For an example, we adopt a nearest-neighbor selection criterion defined in the following.

For each class $y_m \in \mathbb{Y}$, let $\mathcal{X}_m$ and $\hat{\mathcal{X}}_m^{(k)}$ be the subset of labeled data whose class is y_m, and the set of unlabeled samples whose class prediction is y_m, respectively

$$\mathcal{X}_m = \left\{ \boldsymbol{x} \,\middle|\, (\boldsymbol{x},\, y_m) \in \mathscr{L} \right\} \tag{4.7}$$

and

$$\hat{\mathcal{X}}_m^{(k)} = \left\{ \boldsymbol{x} \,\middle|\, \boldsymbol{x} \in \mathcal{U} \text{ and } \hat{g}^{(k)}(\boldsymbol{x}) = y_m \right\} \tag{4.8}$$

We then define the nearest predicted sample from $\hat{\mathcal{X}}_m^{(k)}$ to the labeled class $\mathcal{X}_m$ as

$$\operatorname*{arg\,min}_{\boldsymbol{x} \in \hat{\mathcal{X}}_m^{(k)}} \left(d(\boldsymbol{x}, \mathcal{X}_m) = \frac{1}{|\mathcal{X}_m|} \sum_{\boldsymbol{x}' \in \mathcal{X}_m} d(\boldsymbol{x},\, \boldsymbol{x}') \right) \tag{4.9}$$

where $d(\boldsymbol{x}, \mathcal{X}_m)$ is the distance between a sample $\boldsymbol{x}$ and the set $\mathcal{X}_m$ defined in Equation (4.9) to be the average distance between the sample and all members of the set. In Equation (4.9), $d(\boldsymbol{x}, \boldsymbol{x}')$ is the distance between two data points in the feature space $\mathbb{X}$ and is a design choice. In all our experiments, this choice is the Euclidean distance $d(\boldsymbol{x},\, \boldsymbol{x}') = ||\boldsymbol{x} - \boldsymbol{x}'||_2$.

The nearest-neighbor augmentation set $\mathrm{d}\hat{\mathscr{L}}_k$ is then given by

$$\mathrm{d}\hat{\mathscr{L}}_k = \left\{ \left(\operatorname*{arg\,min}_{\boldsymbol{x} \in \hat{\mathcal{X}}_m^{(k)}} d(\boldsymbol{x}, \mathcal{X}_m),\, y_m \right) \right\}_{m=1}^{M} \tag{4.10}$$

which in words comprises the nearest unlabeled sample to each class set $\mathcal{X}_m$ predicted to have the same class label y_m.

Augmentation: Selected samples $\mathrm{d}\hat{\mathscr{L}}_k$ augment the training set $\hat{\mathscr{L}}_k$ for the next training iteration

$$\hat{\mathscr{L}}_{k+1} = \hat{\mathscr{L}}_k \cup \mathrm{d}\hat{\mathscr{L}}_k \tag{4.11}$$

Stopping: Steps of *training*, *prediction*, *selection*, and *augmentation* are repeated for K iterations. That is, we adopt a fixed number of iteration as a stopping criterion.

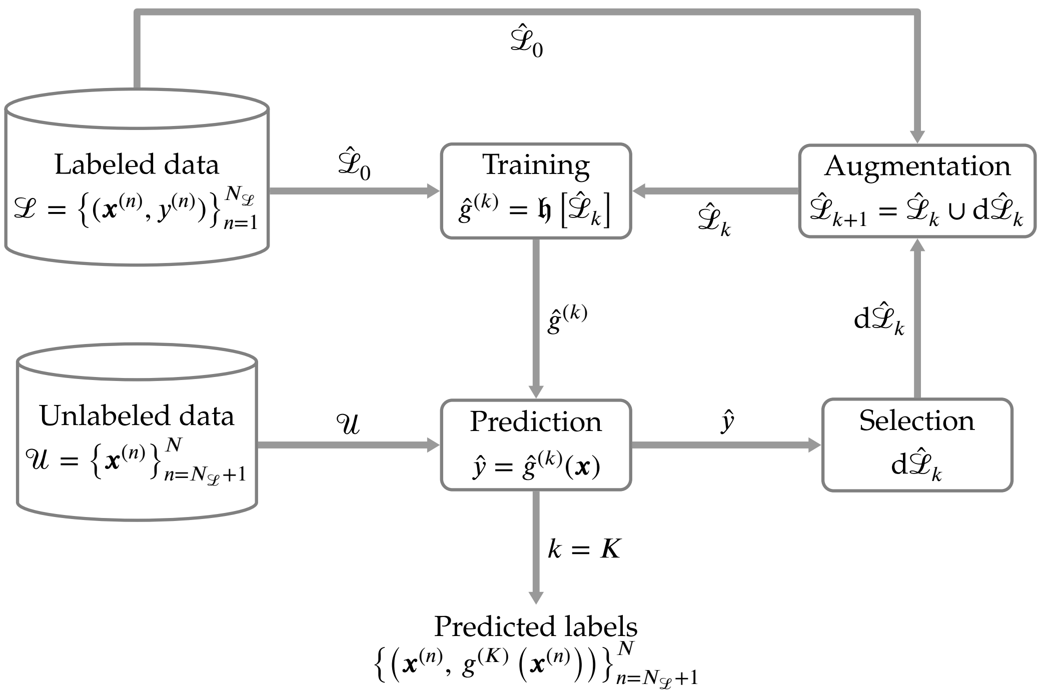

$\hat{\mathscr{L}}_0$
Labeled data
$\mathscr{L} = \{(\boldsymbol{x}^{(n)}, y^{(n)})\}_{n=1}^{N_{\mathscr{L}}}$
$\hat{\mathscr{L}}_0$
Training
$\hat{g}^{(k)} = \mathfrak{h}[\hat{\mathscr{L}}_k]$
$\hat{\mathscr{L}}_k$
Augmentation
$\hat{\mathscr{L}}_{k+1} = \hat{\mathscr{L}}_k \cup \mathrm{d}\hat{\mathscr{L}}_k$
$\hat{g}^{(k)}$
$\mathrm{d}\hat{\mathscr{L}}_k$
Unlabeled data
$\mathscr{U} = \{\boldsymbol{x}^{(n)}\}_{n=N_{\mathscr{L}}+1}^{N}$
$\mathscr{U}$
Prediction
$\hat{y} = \hat{g}^{(k)}(\boldsymbol{x})$
$\hat{y}$
Selection
$\mathrm{d}\hat{\mathscr{L}}_k$
$k = K$
Predicted labels
$\{(\boldsymbol{x}^{(n)}, g^{(K)}(\boldsymbol{x}^{(n)}))\}_{n=N_{\mathscr{L}}+1}^{N}$

Figure 4.1.: Block diagram of a simple self-training system. The output from the system is either estimated labels $\{(\boldsymbol{x}^{(n)}, g^{(K)}(\boldsymbol{x}^{(n)}))\}_{n=N_{\mathcal{L}}+1}^{N}$ (for transductive learning) or a semi-supervised classifier $\hat{g}^{(K)}$ (for inductive learning). Training starts from the seed labeled $\mathscr{L} \equiv \hat{\mathscr{L}}_0$ being fed to the training algorithm $\mathfrak{h}[\cdot]$ for estimating the fully supervised classification function $\hat{g}^{(0)}$. It then proceeds iteratively in the loop (training, prediction, selection, and augmentation) for K iterations.

In Figure 4.1, a visual illustration of the iterative self-training framework adopted in this work is provided. As depicted in the illustration, training is built up on a labeled set and an unlabeled one. As training proceeds, a new classifier is trained on the externally labeled data, and a few samples from the unlabeled set along with the classifier's own prediction on these samples.

4.5. Remarks

One advantage of this approach is its applicability to any learning algorithm (or supervised classification model) as long as a suitable selection criteria can be defined. Empirical evidence, however, shows a better fit to maximum margin classification models. Another advantage is that it does not require additional clustering components to reveal the density (or structural information) in the input space, rather it estimates this structure from its own predictions.

Admittedly, the adopted selection and augmentation steps are vulnerable to error propagation. In other words, if a classier $\hat{g}^{(k)}$ at iteration k results in a false prediction for a set of samples, such mis-predictions have the chance of reaching the augmentation set $d\hat{\mathscr{L}}_k$ and, in turn, the training set $\hat{\mathscr{L}}_{k+1}$ for the following iteration. The following classifier $\hat{g}^{(k+1)}$ will then be trained on a faulty set of labeled data with no assumption of a faulty ground truth. One can, however, adopt various measures to remedy the risk of error propagation. Examples of these measures include reserving a validation subset $\mathscr{V}$ of the original labeled data $\mathscr{L}$ for a test of degradation in performance and potentially early stopping. Another is simply self-training for a relatively low number of iterations K. The latter is our adopted approach where self-training is limited to $K = 3$ iterations in our experiments with a notable boost in performance.

As final remark we note that the unlabeled set $\mathscr{U}$ remains the same for all iterations. As a result, augmentation samples are permitted to change labels in future iterations which is adopted as a counter measure against the problem of error propagation. This is indeed a reasonable choice for different reasons.

The first, as stated earlier, is to avoid error propagation as early augmentation samples are very likely to contain erroneous predictions since the model was trained using smaller labeled sets. As augmentation samples are introduced and with the assumption that SSL is feasible, we are supposed to trust later model predictions rather than earlier ones.

Secondly, the self-training algorithm should give a chance for rare classes to saturate faster than prevalent ones. Removing samples from the unlabeled set $\mathscr{U}$ after each iteration will force rare classes to expand even beyond its boundary region, while stopping very early will leave dense regions poorly exploited. With dynamic augmentation, rare classes are permitted to saturate (that is, by repeatedly selecting the same nearest neighbor) earlier then prevalent ones which will most likely select a new sample each iteration.

A third and final motivation is that, it serves as a stopping criterion for the iterative self-learning procedure. Indeed, if the algorithm ends up in two identical augmentation sets, then this is a clear stopping flag where further iterations are very likely to hurt performance.

4.6. Performance Assessment Measures

Briefly stated, we utilize the macro-F_1-score introduced and defined in Section 2.6 as a final evaluation measure for all experiments.

4.7. Experimental Validation

In validating the proposed model, we tested a self-training classification model base on an multi-class[9] linear SVMs and nearest-neighbor selection criterion on the publicly available BLUED dataset [Anderson et al. 2012].

As mentioned in Section 3.3.1, BLUED [Anderson et al. 2012] is a residential energy dataset comprising aggregate measurements of two phases (phases A and B with the latter containing more loads and wider range of load categories including several miscellaneous appliances) of a household for one week. We mention in this section further details that are mostly relevant to the SSL experimental setups and refer the reader to the original work by Anderson et al. [2012] for a thorough description.

Originally, BLUED contained 872 events in its first phase and 1548 in the second. For appropriate validation, we built transition-level labeling for these events where *on-*, *off-*, and state-change events are distinguished (such information is not provided in BLUED by default). Additionally, we assigned different labels to each load state (e.g. refrigerator's lights). Finally, we neglected some events that were either of unknown circuits or simultaneously occurring with other events. Handling simultaneous events is a task of either a feature extraction stage or a smarter energy estimation component. We ignored these events since our interest at this step is the comparison between two different classification approaches. Such manual cleansing led to a refined set $\mathscr{D}$ of labeled observations consisting of $N^{(A)} = 749$ events

[9]More precisely, a multi-class construction of binary SVMs.

spanning $M^{(A)} = 23$ classes for phase A and $N^{(B)} = 1284$ events spanning $M^{(B)} = 45$ classes for phase B.

Feature vectors are limited to step changes in real and reactive power signals for each event $\boldsymbol{x} = [\texttt{dP}, \texttt{dQ}]^\mathsf{T}$ and are extracted from the post-computed 60 Hz signals using a clustering-based event detection algorithm [Barsim et al. 2014a]. Training and selection utilized the Euclidean distance function $d(\boldsymbol{x}, \boldsymbol{x}') = ||\boldsymbol{x} - \boldsymbol{x}'||$. In all experiments, the adopted self-training model is limited to $K = 3$ iterations .

Two semi-supervised settings are investigated in the sequel, namely, *transductive* and *inductive* learning (see Chapelle et al. [2006] and Zhu and Goldberg [2009] for a detailed discussion and observe the illustration in Figure 4.2 for the next subsections).

4.7.1. Transductive Learning Experiments

A *transductive* setting of SSL is a learning scheme in which both the labeled $\mathcal{L}$ and unlabeled $\mathcal{U}$ sets are exposed to the training framework during its training phase with the objective of predicting labels $\hat{y}$ for the elements in the unlabeled set $\mathcal{U}$ only.

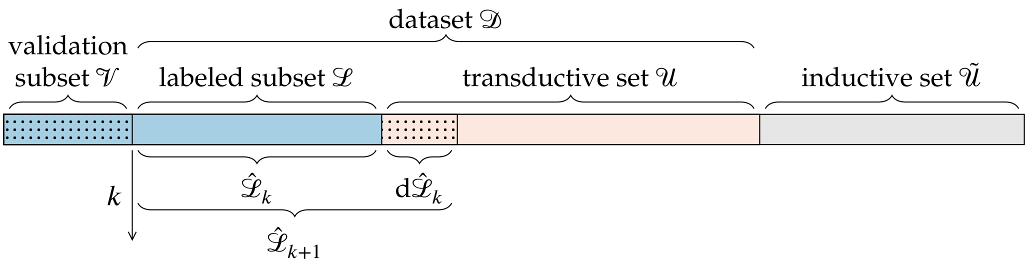

dataset $\mathcal{D}$
validation subset $\mathcal{V}$
labeled subset $\mathcal{L}$
transductive set $\mathcal{U}$
inductive set $\tilde{\mathcal{U}}$
k
$\hat{\mathcal{L}}_k$
$\mathrm{d}\hat{\mathcal{L}}_k$
$\hat{\mathcal{L}}_{k+1}$

Figure 4.2.: Data splits for transductive and inductive settings of SSL. In case of validation-based stopping, a validation subset $\mathcal{V}$ is first reserved for estimating the stopping indicator. The rest of the dataset is split into a labeled subset $\mathcal{L}$ and unlabeled set $\mathcal{U}$. The latter is referred to as the *transductive* set from which augmentation samples $\mathrm{d}\hat{\mathcal{L}}$ are incrementally incorporated in training the base model. An inductive $\tilde{\mathcal{U}}$ set is beyond all samples that have been observed at any step of training. Validation-based early stopping as a counter measure for error propagation can be achieved using the pre-reserved validation set $\mathcal{V}$ but is not our choice in this work.

In other words, transductive learning does not need to generalize beyond the given dataset $\mathscr{D}$, and its main output is a set of predicted class labels $\{(\boldsymbol{x}^{(n)}, \hat{y}^{(n)})\}_{n=N_{\mathscr{L}}+1}^{N}$ rather than a trained classifier $\hat{g}^{(K)}$ as in the case of inductive learning.

Observe in Figure 4.2, for example, how a trainer progressively acquires new samples from the transductive subset, but not the inductive one. A transductive model is one that only evaluates on the transductive set.

Figure 4.3 compares the performance of a traditional supervised model with a self-training one on a growing number of labeled observations $N_{\mathscr{L}}$. The test set $\mathscr{D}$ is divided into two subsets, the labeled subset $\mathscr{L}$ and an unlabeled set $\mathscr{U}$ and both are exposed to the training algorithm. Therefore, the semi-supervised system here is an example of a transductive setting (i.e. the system is permitted to observe all test queries during the learning phase).

Note that the selected labeled set $\mathscr{L}$ in both phases covers the whole label space and therefore starts from one observation per class. Evaluations are taken over a random subset $\tilde{\mathscr{D}} \subset \mathscr{D}$, selected in advance and fixed for all evaluations.

Tests on phase A of BLUED show an noticeable gain in performance particularly in the regions where labeled observations are relatively scarce (that is, below 12% or equivalently 90 events). Once sufficient labeled data is available, the two models perform almost identically. This shows how an SSL system can benefit from unlabeled data in order to recover a loss in performance due to insufficient labeling information.

Phase B shows a gain in the performance as well, but it is rather smaller compared to the first phase. This can be attributed to the weak class separation in the second phase compared to the first one, along with our choice of linear kernel SVM models.

4.7.2. Inductive Learning Experiments

A more challenging setting of SSL is the *inductive* learning scheme. In this setting, the objective of the trained model $\hat{g}^{(K)}$ is to generalize to query observations beyond both subsets $\mathscr{L}$ and $\mathscr{U}$. Therefore, the main output is the classifier itself $\hat{g}^{(K)}$ rather than its predictions on the unlabeled set $\mathscr{U}$. Inductive SSL is intuitively interpreted as an augmentation of the training dataset with unlabeled data for reaching a more robust prediction model.

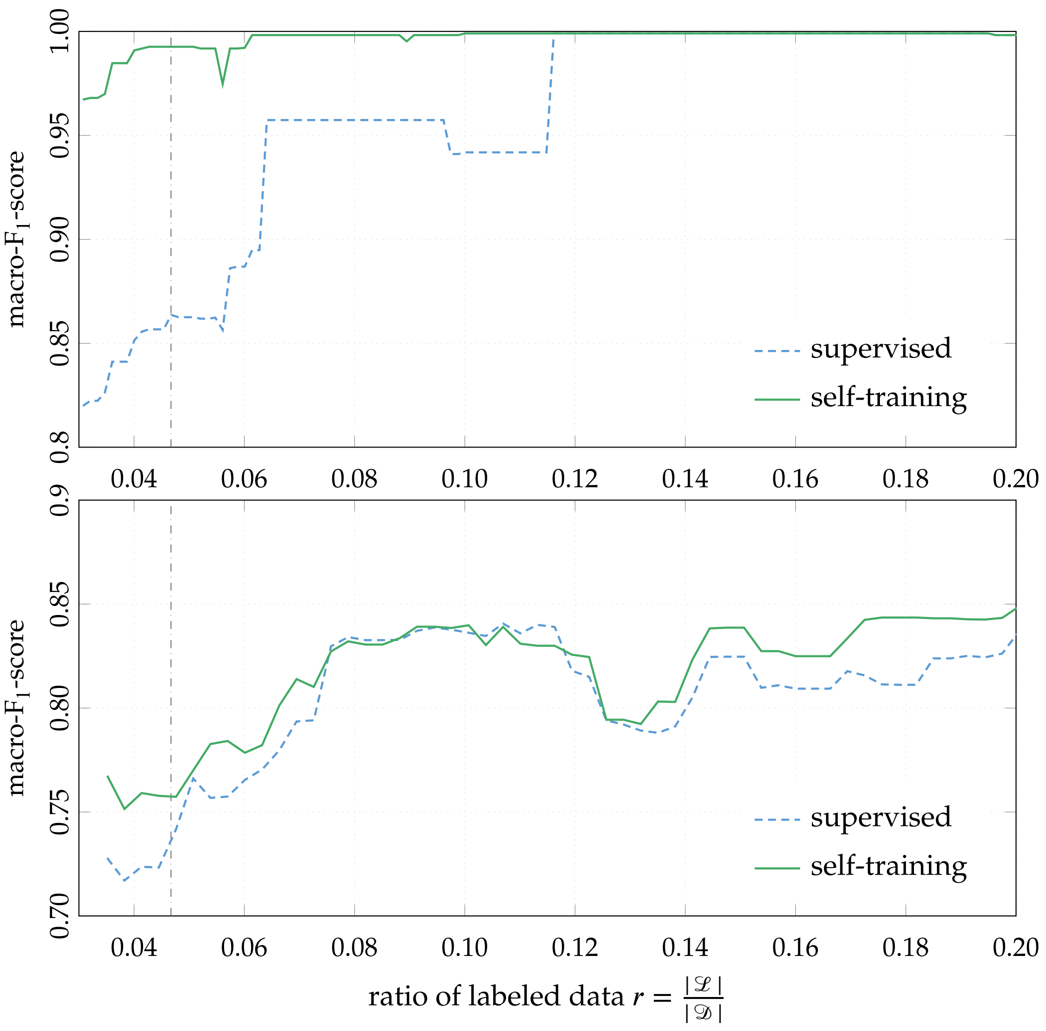


Figure 4.3.: Transductive validation of the proposed self-training model (solid green) compared with a traditional supervised model (dashed blue) when applied to extracted events of phase A (upper) and phase B (lower) of the BLUED dataset. The dash-dotted vertical line at almost 4.67% of labeling is the cut analyzed in Figure 4.4. $r = |\mathscr{L}| / |\mathscr{D}|$ is the ratio of labeled observations. In all cases, we estimated the performance of the output classifier $\hat{g}^{(K)}$ on a randomly selected set of observations. The selected set is fixed for all tests in order to give a fair comparison and it can contain samples from either the labeled set $\mathscr{L}$ or the unlabeled set $\mathscr{U}$.

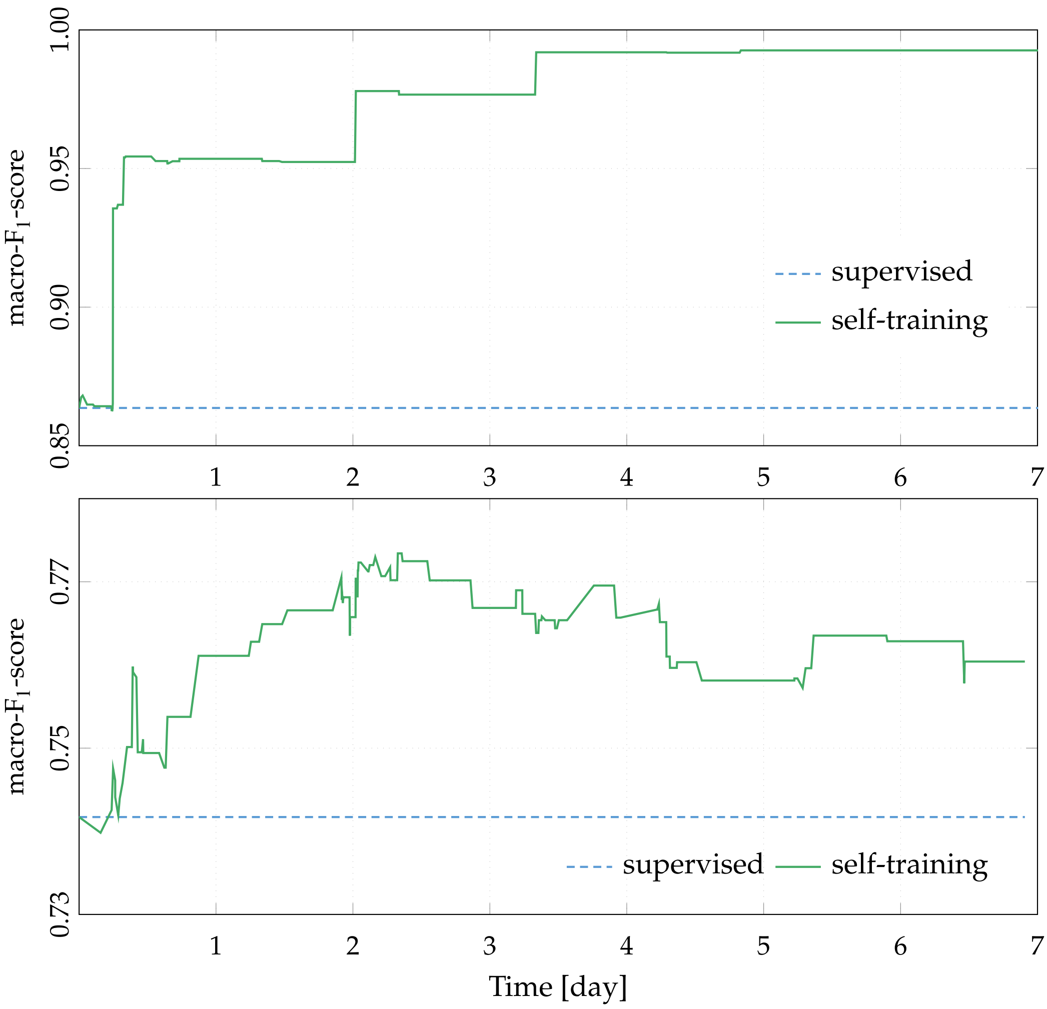


Figure 4.4.: Transductive and inductive validation of the proposed semi-supervised model (solid green) compared to the fully supervised one (dashed blue) over time, with the fully inductive case starting at $T = 0$ days and reaching a fully transductive scheme at $T = 7$ days. Labeled data for both phase A (upper) and phase B (lower) is fixed to 35 and 60 samples, respectively. In all cases, we estimate the performance of the output classifier $\hat{g}^{(K)}$ on a randomly selected set of observations. The selected set is fixed for all tests in order to give a fair comparison and it can contain samples from either the labeled set $\mathscr{L}$, the unlabeled set $\mathscr{U}$, or the remaining samples that are never seen by the learner $\mathfrak{h}$.

Figure 4.4 shows the classification performance of an inductive SSL system over time. In this case, the labeled set is kept constant (and small) for the two phases. For phase A, 35 random samples were labeled while phase B was provided with 60 labeled samples (both constitute almost 4.67% of the total number of events in that phase or almost 8 hours of labeling). A traditional supervised model is dependent only on the labeled set and since the labeled set is fixed and the adopted model (i.e. SVM) provides a unique solution, the performance of the traditional system remains constant all the time.

On the contrary, a semi-supervised system leverages both labeled and unlabeled observations. It starts from almost the same performance of a traditional supervised model (fully inductive far left in the figure). Afterwards, the system keeps observing new unlabeled events (represented in the incrementally growing set $\mathcal{U}$ and reaching the fully transductive setting far right), deduces more internal structural information, and attempts to estimate a better classifier $\hat{g}^{(K)}$.

Both figures show an increasing performance of an semi-supervised system over time. However, phase B shows less stable behavior and also less gain. Once again, this is hypothesized to the relatively well-separated classes in phase A compared to phase B.

4.8. Conclusion

In this work, we introduced a simple semi-supervised classification tool to an energy disaggregation system in order to reclaim the loss in performance due to labeled data scarcity. The proposed models empirically shows performance gains via leveraging unlabeled observations and, therefore, reduces labeling costs and efforts.

Despite its simplicity, the algorithm was able to provide a learning energy disaggregation system that leverages both external information in the form of hand-labeled observations and internal structural information revealed from the unlabeled detections. For batch processing or offline deployments (that is, the transductive learning setup), a semi-supervised system reduces the amount of labeling effort required. For an online system, on the other hand, and in case the model is permitted to fine-tune its parameters post-deployment, the proposed model results in a system that increases performance over time as it observes more unlabeled instances at no extra cost.

Future work should investigate the choice of a probabilistic base model capable of providing meaningful uncertainty scores. Additionally, self-training is one of the simplest SSL paradigms that yet showed promising results on a real-world energy dataset. More stable and profound SSL paradigms are expected to show even superior performance and remedies to data scarcity and should, therefore, be a topic of future research.

Finally, we give a word about practicality of SSL settings in general and the discussed self-training paradigm in particular for energy disaggregation. As discussed throughout this chapter, SSL is mostly beneficial in situations where labeled data are scarce while unlabeled observations are abundantly available. This notably occurs in two forms of energy monitoring deployments. The first is the long-term slow variations in the target building resulting from, as an example, newly introduced loads, seasonal shifts, and end-users' habitual changes. In this case, energy monitoring definitely benefits from SSL in rapidly adapting to the new changes at minimal costs. For instance, the self-training paradigm would require a few labels of the newly introduced load for it to get incorporated in the monitoring process.

The second case is an abrupt change in the target deployment such as generalizing to new buildings. Numerous aspects determine the benefit of SSL in this scenario. For example, if the new building shares a considerable set of end-use loads (e.g. in the case of generalizing across households within a close geographical proximity) and especially if variations in loads' signatures are minimal, then SSL is expected to effectively mitigate the burden of labeled data acquisition. However, SSL is not expected to be effective if the domain change suffers from a considerable shift in the feature space. In this latter scenario, transfer learning algorithms should be leveraged instead.

Chapter 5.

Generic Deep Disaggregation

In Chapter 3, the significance of a well engineered feature set was particularly emphasized for efficient clustering (or similarly classification in supervised energy disaggregation frameworks) and more accurate and robust energy estimation. This, in turn, introduced further complexities and imposed higher requirements on the early stages of either time series analysis or change detection and segmentation.

An alternative framework, known as the deep learning framework, is to optimize such a multi-stage system (e.g. preprocessing, feature extraction, classification ... etc) jointly using a common objective function under gradient-based optimization [LeCun et al. 1998a] and utilizing the back-propagation algorithm [Rumelhart et al. 1986, LeCun et al. 1998b] that is supported by automatic differentiation tools [Baydin et al. 2018]. In this chapter, we review prior art (to the date of our publication [Barsim and Yang 2018]) in energy disaggregation that utilized deep learning frameworks for such a task, and introduce a well regularized deep black-box model based on convolutional neural nets suitable for generic end-to-end load monitoring and disaggregation.

This chapter is organized as follows. In the first section we introduce how the deep learning framework applies to the energy disaggregation pipeline, and review prior art on this topic in the following section. The remainder of this chapter presents a study on leveraging the deep learning framework on the problem of energy disaggregation. We first state the addressed problem in Section 5.3. The proposed model is detailed in Section 5.4 along with our modeling assumptions. Performance assessment measures are defined and discussed in detail in Section 5.5. Section 5.6 reports and discusses our main results on one of the freely available energy datasets, and Section 5.7 summarizes and concludes this chapter.

Work in this chapter was published by Barsim and Yang [2018], and phrasing, figures, and results are partially taken from our prior work. Moreover, literature review is limited to works preceding our aforementioned publication.

5.1. Introduction

Deep learning is a machine learning framework in which a multi-stage system (such as the energy disaggregation pipeline) is jointly optimized based on an end-to-end objective function and using gradient-based optimization [LeCun et al. 1998a], which in turn leverages the power of the gradient backward propagation (known commonly as *back-propagation* or *reverse accumulation*) algorithm [Rumelhart et al. 1986] implemented in modern automatic differentiation tools [Baydin et al. 2018]. Such models are normally over-parameterized in order to feature high modeling capacities allowing for end-to-end learning from an abundance of training examples. The hierarchical architecture of deep models (aka depth) is a key component that allows for progressive abstraction of data from low level processing in early stages or layers (e.g. filtering, translation, rotation ... etc) to high level abstractions in later ones.

Especially in the last decade and due to numerous factors, deep learning has been widely applied to an exceptional variety of machine learning problems and domains such as, and most notably, computer vision [Krizhevsky et al. 2012, Ciresan et al. 2012], dynamic system modeling [Yeo and Melnyk 2019, Längkvist et al. 2014, Duncker et al. 2019], reinforcement learning [Jeerige et al. 2019, Caicedo and Lazebnik 2015], natural language processing [Mikolov et al. 2013, Socher et al. 2012], graphical data modeling [Santoro et al. 2017, Defferrard et al. 2016], healthcare and medical diagnostics [Zhang et al. 2019, Khuriwal and Mishra 2018], context-aware data compression [Goyal et al. 2019, Bégaint et al. 2019], acoustic and speech applications [Ling et al. 2015, Trigeorgis et al. 2016, Miao et al. 2015], remote sensing [Zhang et al. 2016], to name but a few. A detailed review of deep learning approaches, applications, and the history of deep learning is outside the scope of this work but we refer the interested reader to some of the ubiquity of reviews on the topic such as [Schmidhuber 2015, Bengio et al. 2013, Shrestha and Mahmood 2019, Modi 2018, Wan 2019] and [Guo et al. 2016].

Viewed as a pipeline of traditional machine learning stages (raw data processing, feature extraction, clustering, energy estimation ... etc), energy disaggregation becomes one of the target applications for the deep learning framework due to its progressive learning design. For example, it is common in various energy monitoring and disaggregation approaches to begin with first order differences for the detection of abrupt changes. Higher order differences (e.g. distances between event-based features) are utilized in later stages such as event-level clustering or classifications. Such a hierarchical architecture, along with the exceptional success in many other applications, motivated research on deep disaggregation models.

In this chapter, we discuss the feasibility of utilizing deep learning as a black-box modeling framework for energy disaggregation (what we refer to as a generic deep disaggregation framework). Our claim is that, *a carefully regularized deep disaggregation model with sufficient modeling capacity is able to learn the (short-term) load profiles without the need for load-dependent fine tuning nor complete knowledge of the target environment*.

It should be noted, however, that we do not devalue domain knowledge, hybrid modeling, nor neural architecture search approaches for deep energy disaggregation. We, rather, hope to answer in this work the question of whether or not a generic load-independent deep disaggregation model is feasible once an adequate amount of training data has been obtained, and how it compares to load-tuned disaggregation models.

5.2. Related work

In this section, we briefly review some of the most recent works (to the date of our publication [Barsim and Yang 2018]) on energy disaggregation and load monitoring that adopted data-driven learning techniques, or more precisely the deep learning framework. We limit this review to those works that followed the recent breakthrough in data-driven learning with neural architectures [Krizhevsky et al. 2012] which proved to be remarkably powerful in various machine learning and data mining applications, more prominently in cases where adequate (or even an abundance of) observations are available with limited or missing domain knowledge.

Mauch and Yang [2015] exploited a generic two-layer bidirectional recurrent neural neural (RNN) architecture featuring long short term memory (LSTM) [Hochreiter and Schmidhuber 1997] recurrent units in extracting single load profiles. They tested their models on the reference energy disaggregation dataset (REDD) [Kolter et al. 2010] in a de-noised scheme[1] [Makonin and Popowich 2015]. Additionally, they validated the generalization of their architecture to previously unseen loads in new buildings. In a later work, Mauch and Yang [2016] used a combination of discriminative and generative models in a two-stage eventless extraction of load profiles.

Kelly and Knottenbelt [2015a] evaluated and compared three neural network architectures on domestic loads from the UK-domestic appliance level energy (UK-DALE) [Kelly and Knottenbelt 2015b]. The first is a bidirectional RNN architecture with LSTM units similar to the one adopted by Mauch and Yang [2015]. The second follows the architecture of a de-noising auto-encoder (dAE) [Vincent et al. 2010]. Finally, the third model is a regression-based disaggregator whose objective is to estimate the main key points of an activation cycle of the target load within a given window. Such key points include the start-time, end-time, and average energy consumed in the first activation cycle in the given window.

Similarly, He and Chai [2016] applied two architectures, namely a convolutional dAE and an RNN, to the same problem. In their architectures, they applied parallel convolutional layers with different kernel sizes analogous to the Inception module introduced in GoogLeNet [Szegedy et al. 2015]. Zhang et al. [2018] simplified the objective of the dAE architecture in [Kelly and Knottenbelt 2015b] to predict a single time instance of the target load profile for a given window of the aggregate signal. Their work is inspired by similar approaches in related application domains such as speech signal processing. Likewise, Nascimento [2016] applied three neural network architectures, namely basic convolutional dAE, an RNN, and a ResNet-based model [He et al. 2016a] to the same problem but on three target loads in the REDD dataset [Kolter et al. 2010]. He introduced several improvements such as redefin-

[1]A scheme in which aggregate signals are synthesized from the measured segregate profiles. In this scheme, all information about underlying loads is known, and the only source of noise is sensor or measurement noise.

ing the loss function, exploiting batch normalization [Ioffe and Szegedy 2015], and applying residual connections [He et al. 2016a].

Additionally, Lange and Bergés [2016] adopted a deep neural network with constrained binary and linear activation units in the last two layers. Their first objective was to retrieve relevant sub-components of the input signal that sum up linearly to the aggregate active and reactive powers. Subsequently, they estimate the *on-off* activation profile of each load. Notable yet is that their approach was applied on the very-high frequency (12 kHz) current and voltage measurements from the BLUED dataset [Anderson et al. 2012].

Contrary to these event-less approaches, Kaman et al. [2017] applied a basic multi-layer perceptron (MLP) [Bishop 2006] in an event-based setting. The model is used in the event classification stage adopted in event-based monitoring systems whose input is the feature vector extracted from detected events[2]. The study was, however, performed in a de-noised setup and limited to merely two loads.

In many of the aforementioned works [Kelly and Knottenbelt 2015b, He and Chai 2016, Zhang et al. 2018, Nascimento 2016], each disaggregation model is a neural network whose disaggregation window length (and consequently the width of subsequent layers) depends on the load being monitored. The disaggregation window of each load is manually adjusted in a per-load basis to fully capture a single activation cycle of the load, and its width is treated as a load-dependent hyper-parameter affecting both training and deployment. Moreover, the disaggregation performance widely differs amongst variant load categories and a model that achieves remarkably well on one load might drastically fail for other loads.

This section represents a nearly comprehensive review of related work (to the date of our publication [Barsim and Yang 2018]) on energy disaggregation using data driven learning which supports our claim that joint progress in the two fields (energy disaggregation and data-driven learning) lags notably behind the development of each field individually. Therefore, one of the objectives of this chapter is to assert the feasibility of data-driven models in the problem of energy disaggregation if the adopted model possesses adequate capacity and its parameters are used efficiently, in a hope to close this gap.

[2]Abrupt changes in the observed aggregate signal.

5.3. Problem Statement

The problem we address in this chapter is *single load extraction of activation profiles*. Specifically, *can we infer the operating intervals of a certain load given the aggregate load profile ? and how much historical (or future) measurement samples would be required for this task ?*.

In the following subsections, we first introduce what is referred to as *activation profiles* and how they are estimated from segregate load profiles. Afterwards, we formulate the *single load extraction* problem and how the input and target signals are modeled for the given task.

5.3.1. Activation Profiles: Definition and Motivation

In its simplest form, a load is modeled as a finite state machine (FSM) consisting of merely two states, namely, the positive- or *on*-state whenever the load is consuming energy from the main power source, and the negative- or *off*-state otherwise. Accordingly, the load monitoring problem reduces to a set of binary classification tasks as shall be discussed shortly.

Note that unlike prior effort, the consumption profile of a load during its *on*-state need not be a constant nor a piecewise-constant function in time. In other words, previous works defined an *on-off* or a multi-state load model as a finite state machine with constant [Zeifman and Roth 2011] or a piece-wise constant [Makonin 2014] energy consumption level between state-change events, respectively.

In contrast, our proposed model incorporates all load categories that can be, at least once in a while, disconnected from the monitored circuit (that is, switched-*off*) regardless of the complexity of its consumption profile during operation[3]. Such loads are often the main target in load monitoring and disaggregation systems [Liang et al. 2019].

The desired signal (aka ground truth) of the mth load is the discrete-time binary-valued signal $y^{(m)}(t) \in \{0, 1\}$ which is set (i.e. to indicate an *on*-state) at time t when-

[3]Excluded loads are limited to permanently operating ones such as alarm systems and stand-by loads.

ever the load is operating in one of its activation states at that time instance and unset otherwise. In this work, we refer to this signal as the *activation profile*.

Activation profiles have been addressed in [Kelly and Knottenbelt 2015a] but either in a regression-based formulation (which estimates only the change points of the activation profile $y^{(m)}(t)$) or as a post-processed output of the disaggregation model. In this work, activation profiles are the direct output of each disaggregator.

For notational convenience, we will drop the load index $\cdot^{(m)}$ for the rest of this chapter since we only target a single load per model. Nevertheless, load indices may still arise occasionally if deemed necessary to avoid confusion.

Applications that benefit from activation profiles include mainly activity monitoring and occupancy detection [Batra et al. 2015] in which time-of-use information dominates the value of energy consumption. Moreover, activation profiles simplify the disaggregation process and permit the use of a two-stage detect-and-classify model that first detects the activation periods of a load from the aggregate data and then estimates the energy consumed or the fine-grained consumption profiles within each activation cycle. In addition, these profiles facilitate the comparison between different disaggregators of various load types. This is because binary evaluation metrics are normally more easily interpreted than other disaggregation metrics that remain contentious within the energy disaggregation community. In this chapter, only the first stage of estimating the activation profile is considered.

5.3.2. Activation Profiles: Estimation

While some energy datasets, such as Belkin's dataset [Belkin 2013], or UK-DALE [Kelly and Knottenbelt 2015b] (except for e.g. BLUED[4] [Anderson et al. 2012]) include usage data, such information is either unreliable due to manual logging by end-consumers or only available for user-activated loads such as lights, kitchen appliances, washers, cleaners, etc. For that reason, we define the true activation profile of a load $y(t)$ via a threshold-based approach applied to the sub-metered real power

[4]In the case of BLUED [Anderson et al. 2012], switching times are of sufficient reliability for both user-activated and background loads.

signals. This estimation approach is similar to the one used in [Kelly and Knottenbelt 2015a] and implemented in the open-source energy disaggregation library NILM-TK [Batra et al. 2014a] which is defined as follows.

The sub-metered real-power[5] $\mathtt{P}(t)$ of a load is compared against predefined load-dependent thresholds to detect its operation intervals. In order to avoid anomalies and false activations or deactivations, the load is assumed to be in an activation state (i.e. *on*) if its power draw $\mathtt{P}(t)$ exceeds a load-dependent threshold $\mathtt{P}_{\text{on}}$ for a minimum period of time T_{on}. Similarly, if the power draw drops below a predefined threshold $\mathtt{P}_{\text{off}}$ for a given period T_{off}, the load is assumed to be disconnected. Otherwise, the load keeps its last observed state. Thus, the estimated activation profile is defined as

$$y(t) = \begin{cases} 1, & \text{if} \quad \mathtt{P}(\tau) \geqslant \mathtt{P}_{\text{on}} \quad \text{for} \quad \left(t - T_{\text{on}}\right) < \tau \leqslant t \\ 0, & \text{if} \quad \mathtt{P}(\tau) \leqslant \mathtt{P}_{\text{off}} \quad \text{for} \quad \left(t - T_{\text{off}}\right) < \tau \leqslant t \\ y(t-1), & \text{otherwise} \end{cases} \tag{5.1}$$

with the initial state assumed to be *off* (i.e. $y(1) = 0$) for all loads. Note that $\mathtt{P}_{\text{on}}$, T_{on}, $\mathtt{P}_{\text{off}}$, and T_{off} are the only load-dependent parameters in this work, and they are used merely in estimating the ground truth signals. Values of these parameters are similar or close to those adopted in [Kelly and Knottenbelt 2015a] and are listed in Table 5.1 for the sake of completeness.

It is also worth noting that we compared this basic threshold-based technique to a one based on $\underline{\text{h}}$idden $\underline{\text{M}}$arkov $\underline{\text{m}}$odels (HMM) in estimating the ground truth signals [Mauch et al. 2016]. In the latter, a multi-state HMM with normally distributed emissions was trained on the sub-metered load signals where all states with a non-zero mean were assumed to be *on*-states. The number of states in each model is load-dependent and it is estimated so as to minimize the $\underline{\text{A}}$kaike $\underline{\text{i}}$nformation $\underline{\text{c}}$riterion (AIC) which is a function of both the number of states in the model and the likelihood of the observed sequence $\mathtt{P}(t)$. However, we found that the complexity of this approach did not match its added value and was, therefore, excluded from

[5]We restrict all experiments in this chapter to the real power signals for both aggregate and segregate data for computational advantages, ease of deployment, and fair comparisons with prior art. However, involving additional input channels to the adopted models should be straightforward with minimal changes to the proposed model architecture.

Table 5.1.: Load-dependent thresholds for estimating the ground truth signals (i.e. activation profiles) of different loads from their sub-metered real power signals. In all experiments, the *on*- and *off*-power thresholds are identical. Moreover, T_{on} and T_{off} are sample counts in eq. (5.1) but their values are given in this table in minutes.

Load	$\mathtt{P}_{\text{on}} = \mathtt{P}_{\text{off}}$ [Watt]	T_{on} [min]	T_{off} [min]
Fridge (FR)	5	1	1
Lights (LC)	10	1	1
Dishwasher (DW)	10	30	5
Washing machine (WM)	20	30	5
Solar thermal pump (SP)	20	1	1
Television (TV)	5	3	3
Boiler (BL)	25	5	5
Kettle (KT)	1000	1/3	1/6
Microwave (MC)	50	1/6	1/6
Toaster (TS)	300	1/6	1/20

this writing. In addition, HMM-based modeling was not applicable to some uncontrolled and variable-consumption loads such as lighting circuits and personal computers as reported in our previous work [Mauch et al. 2016].

5.3.3. Single load extraction

In single-load extraction, each disaggregator targets exclusively a single load in the monitored circuit and normally ignores dependencies between loads. We shall discuss this assumption in the following subsection.

Let $x(t)$ denote the discrete-time aggregate signal utilized for the activation profile extraction. In our experiments, the aggregate signal is the whole-house real power profile[6] $x(t) = \mathtt{P}(t)$. Let further, $\boldsymbol{x}_{a:b}$ where $a < b$ denote a finite sub-sequence of the

[6]Note that $\mathtt{P}(t)$ denotes the aggregate real signal not the real power profile of the mth load which would have been denoted in this context by $\mathtt{P}^{(m)}(t)$ to avoid confusion.

signal $x(t)$ from the time instance $t = a + 1$ to $t = b$ given by

$$\boldsymbol{x}_{a:b} = \Big[x(a+1), \quad x(a+2), \quad \cdots, \quad x(b)\Big]^{\top}$$

and likewise for $\boldsymbol{y}_{a:b}$ (revise Section 1.6).

Formally, we define *single load extraction* as the process of inferring the activation profile $\boldsymbol{y}_{a:b}$ of a particular target load in the time interval $a < t \leqslant b$ having merely observed the aggregate signal within the same interval[7] $\boldsymbol{x}_{a:b}$.

In the following, and without loss of generality, we limit our discussion to the time interval $0 < t \leqslant T$ where $T \in \mathbb{N}$ such that input aggregate signal and target activation profile are denoted by $\boldsymbol{x}_{0:T}$ and $\boldsymbol{y}_{0:T}$, respectively. Later, the interval length T becomes a design hyper-parameter and shall be discussed in Section 5.4.6.

5.3.4. The Assumption of Independent Loads

As stated in the preceding sub-section, single load extraction implicitly implies independent loads given the aggregate load profile $\boldsymbol{y}^{(m)}_{0:T} \perp\!\!\!\perp \boldsymbol{y}^{(m')}_{0:T} \mid \boldsymbol{x}_{0:T}$ such that

$$p\left(\boldsymbol{y}^{(m)}_{0:T} \mid \boldsymbol{y}^{(m')}_{0:T}, \boldsymbol{x}_{0:T}\right) = p\left(\boldsymbol{y}^{(m)}_{0:T} \mid \boldsymbol{x}_{0:T}\right) \quad \forall m \neq m' \tag{5.2}$$

where $\boldsymbol{y}^{(m)}_{0:T}$ and $\boldsymbol{y}^{(m')}_{0:T}$ are the activation profiles of the m^{th} and $(m')^{\text{th}}$ loads, respectively, and $(\,\cdot \perp\!\!\!\perp \cdot \mid \cdot)$ denotes conditional independence. One can intuitively question the validity of this assumption, but for this work, we consider it a simplifying assumption for ease of disaggregation and generalization.

While exploiting loads' dependencies is expected to improve the performance of a disaggregation system in a given building [Kim et al. 2010, Kolter et al. 2010, Makonin 2014], it is also likely to reduce the generalization capability of such a system to new, previously unseen buildings. This is because such dependencies originate not only from the physical architecture of the power line network and the assumed signal model but also from the usage behavior of end-consumers which varies widely from one building to another, especially within the residential sector [Batra et al. 2014b].

[7]Admittedly, this is a limitation of this work, and we discuss further possibilities in Section 5.7.

Additionally, the assumption of independent loads permits load monitoring under partial knowledge (e.g. incomplete knowledge of all existing loads and their signatures) of the target environment. This is particularly useful in applications targeting a subset of load categories. Examples of such applications include energy conservation targeting major loads, activity monitoring targeting consumer activated small appliances, and energy efficiency targeting specific load brands.

We additionally note that one must practice extreme care when modeling load dependencies since spurious dependencies might arise preventing reliable generalization. For example, consider a two-load circuit comprising e.g a dishwasher and a fridge (see for instance the illustrative example in Figure 1.1). From domain knowledge, we know that the former is user-activated while the latter is mostly thermostatically controlled. In other words, we know that these two loads must be *almost* independent and either load can barely indicate activation or deactivation of the other. But since these are the only two loads in the circuit, inferring one (e.g. fridge) having observed the aggregate signal renders the second completely identifiable, that is up to measurement noise. Careless modeling would mistakenly conclude a strong dependence between the loads. However, this model will entirely fail once introduced to new circuits, or even when new loads are introduced to the monitored circuit, since the previously observed dependence would no longer be valid. While this is an oversimplified example, we elaborate on our claim more rigorously in the following.

To illustrate and formally state our claim, Figure 5.1 depicts a graphical representation of the causal structure of end-use loads in a household. In the graph we assume that loads do not directly cross talk[8], and a load state can only be changed based on the user intention. Therefore, the user intention H is the direct, but latent, cause of any load profile $y(t)$ (depicted in the graph as a directed edge) and each load profile is one cause of the aggregate whole-house profile $x(t)$. Under these assumptions, there are two sources for inter-load dependencies to appear.

The first is conditioning on the common effect, that is, the aggregate signal. In this case, loads *appear* to be dependent while in fact they are not. A model should not rely on this dependence since it does not reflect the real cause (nor dependence)

[8]An assumption that is very likely to hold in many existing households, but may fail in modern home automation (or internet-of-things (IoT)) architectures.

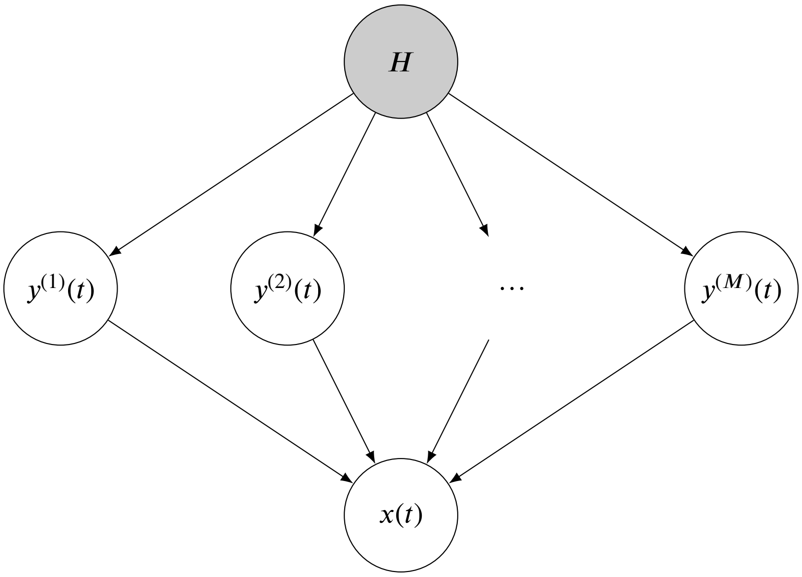


Figure 5.1.: A graphical model illustrating potential dependencies across loads where H is a hidden variable (grayed out) representing user intention [Yang and Zhou 2011], $y^{(m)}(t)$ is the mth load profile, and $x(t)$ is the aggregate signal. Assuming loads do not intercommunicate (e.g. no load causes another load to activate or deactivate) two scenarios may result in load dependencies. The first is conditioning on the aggregate signal $x(t)$ (the effect) which results in spurious dependencies between loads and we claim that a model should not generalize based on this dependency. The second case results from ignoring (that is, not adjusting for) the user intention H (the cause) and such a dependency is definitely informative but hardly measurable.

between loads, and will dramatically change as new loads are introduced (in other words new parents of $x(t)$ are added to the graph).

The second case is not adjusting for (i.e. not conditioning on) the user intention [Yang and Zhou 2011], that is, the common cause between load profiles. This results in real dependencies reflecting user intention and inferring consumers' behavior. While this is a more promising load dependency to investigate, we hardly found prior art in this direction (except perhaps [Yang and Zhou 2011]) and causal modeling for energy disaggregation is clearly lagging behind the state-of-art research on causal learning. Adding new loads is less likely to change this relation between loads since under the assumption of no direct cross talk between loads, there would be no open path[9] through other loads to change this relation. For an example, a user whose daily habit is to use a toaster for breakfast and a dishwasher in the evening will not deviate much from this habit should he buy a printer.

The trap here is that observational data alone, no matter how abundant, can not reveal which of these scenarios is the real explanation for observed dependencies. Accordingly, a data-driving model with only access to data (that is, without the underlying model) will remain clueless whether to consider or ignore each observed dependency.

Learning load dependencies is outside the scope of our work, and we assume independent loads conditioned on the aggregate signal to facilitate modeling, avoid learning spurious dependencies, and enable targeted load monitoring and disaggregation. Nevertheless, we definitely encourage future research in this direction.

5.4. Deep Disaggregation Model

In this section, we introduce the proposed model in detail starting from the adopted modeling assumptions in Section 5.4.1. Afterwards, Section 5.4.2 introduces the adopted model architecture with each elementary operation defined in Section 5.4.3 and training procedure detail in Section 5.4.4. Section 5.4.5 defines the predictive function used upon deployment with final remarks and deployment considerations discussed in Section 5.4.6.

5.4.1. Modeling Assumptions

Single load extraction can be cast as a sequence-to-sequence modeling problem, where the input sequence is the aggregate real power $x(t)$ while the output sequence is the binary-valued activation profile $y(t)$.

[9]An open path between two variables in the presented model is a path from a source node A to a target node B with no intermediate node with both edges pointing inwards (for instance, the path $A \cdots \rightarrow C \leftarrow \cdots B$ is blocked by the node C).

First, we split the input and output sequences into non-overlapping sub-sequences of identical length such that the training dataset $\mathscr{D}$ becomes

$$\mathscr{D} = \left\{ \left(\mathfrak{T}_{nT} \boldsymbol{x}_{0:T}, \, \mathfrak{T}_{nT} \boldsymbol{y}_{0:T} \right) \right\}_{n=0}^{N-1} \tag{5.3}$$

where $\mathfrak{T}_{\tau}(\cdot)$ denotes the translation or time-shift operator defined in Equation (1.5), T is the length of each sub-sequence, and N denotes the number of training sub-sequences. The total length of the training profile is given by NT.

We then assume that all sub-sequences are identically distributed and conditionally independent given the aggregate signal (i.i.d. assumptions) such that

$$p\left(\boldsymbol{y}_{0:NT} \mid \boldsymbol{x}_{0:NT}\right) \;=\; \prod_{n=0}^{N-1} p\left(\mathfrak{T}_{nT} \boldsymbol{y}_{0:T} \mid \boldsymbol{x}_{0:NT}\right) \tag{5.4}$$

which is a reasonable assumption, but remains infeasible to model since the considered input sequence $\boldsymbol{x}_{0:NT}$ is prohibitively long. For an impression, in our second experiment (see Section 5.6.3) and for all load types, T comprises a 3-hour sub-sequence in 1 Hz signals (or equivalently $T = 10800$), and the total length NT spans one month of training data (that is, $N = 248$). Therefore, we further simplify the modeling problem with a stronger independence assumption given by

$$p\left(\boldsymbol{y}_{0:NT} \mid \boldsymbol{x}_{0:NT}\right) \;=\; \prod_{n=0}^{N-1} p\left(\mathfrak{T}_{nT} \boldsymbol{y}_{0:T} \mid \mathfrak{T}_{nT} \boldsymbol{x}_{0:T}\right) \tag{5.5}$$

which implies that all information that is needed to infer any sub-sequence of the activation profile $\mathfrak{T}_{\tau} \boldsymbol{y}_{0:T}$ is encoded in the synchronous input signal $\mathfrak{T}_{\tau} \boldsymbol{x}_{0:T}$ for any non-negative integer $\tau \in \mathbb{Z}_{\geqslant 0}$. We further elaborate on this assumption in Section 5.4.6 but also encourage future investigation of the validity and effect of this simplifying assumption.

Additionally, within each sub-sequence we impose once more the i.i.d. assumptions on each $y(t)$ within the sub-sequence $\mathfrak{T}_{\tau} \boldsymbol{y}_{0:T}$ such that

$$p\left(\mathfrak{T}_{\tau} \boldsymbol{y}_{0:T} \mid \mathfrak{T}_{\tau} \boldsymbol{x}_{0:T}\right) = \prod_{t=1}^{T} p\left(\mathfrak{T}_{\tau} y(t) \mid \mathfrak{T}_{\tau} \boldsymbol{x}_{0:T}\right) \tag{5.6}$$

for any non-negative index τ so that the likelihood function of the whole activation profile $\boldsymbol{y}_{0:NT}$ is given by

$$p\left(\boldsymbol{y}_{0:NT} \,|\, \boldsymbol{x}_{0:NT}\right) = \prod_{n=0}^{N-1} \prod_{t=1}^{T} p\left(\mathfrak{T}_{nT}\, y(t) \,|\, \mathfrak{T}_{nT}\boldsymbol{x}_{0:T}\right) \tag{5.7}$$

Our third assumption is on the distribution of the conditional $\mathfrak{T}_{nT}\, y(t) \,|\, \mathfrak{T}_{nT}\boldsymbol{x}_{0:T}$ where we impose a Bernoulli distribution whose parameter $q_{n,t}$ is estimable from the corresponding input sub-sequence $\mathfrak{T}_{nT}\boldsymbol{x}_{0:T}$ via a parameterized deterministic mapping denoted by $\boldsymbol{g}(\,\cdot \,|\, \boldsymbol{\theta}_g)$ so that

$$p\left(\mathfrak{T}_{nT}\, y(t) \,|\, \mathfrak{T}_{nT}\boldsymbol{x}_{0:T},\, \boldsymbol{\theta}_g\right) = \text{Bernoulli}\left(q_{n,t}\right) \tag{5.8}$$

$$= \text{Bernoulli}\left(g_t\left(\mathfrak{T}_{nT}\boldsymbol{x}_{0:T} \,|\, \boldsymbol{\theta}_g\right)\right) \tag{5.9}$$

where $g_i(\,\cdot \,|\, \boldsymbol{\theta}_g)$ denotes the i^{th} output of the vectorial function $\boldsymbol{g}(\,\cdot \,|\, \boldsymbol{\theta}_g)$ and likewise for $q_{n,t}$ and its vectorial counterpart $\boldsymbol{q}_n$. The structure of this function $\boldsymbol{g}(\,\cdot \,|\, \boldsymbol{\theta}_g)$ is the discussion of the following section. Relevant here is the distribution's $\underline{\text{p}}$robability $\underline{\text{m}}$ass $\underline{\text{f}}$unction (PMF) which is accordingly given by

$$p\left(\mathfrak{T}_{nT}\, y(t) \,|\, \mathfrak{T}_{nT}\boldsymbol{x}_{0:T},\, \boldsymbol{\theta}_g\right) = \left(q_{n,t}\right)^{\mathfrak{T}_{nT}\, y(t)} \left(1 - q_{n,t}\right)^{1-\mathfrak{T}_{nT}\, y(t)} \tag{5.10}$$

with the complete likelihood function having the form

$$p(\boldsymbol{y}_{0:NT} | \boldsymbol{x}_{0:NT}\, \boldsymbol{\theta}_g) = \prod_{n=0}^{N-1} \prod_{t=1}^{T} \left(q_{n,t}\right)^{\mathfrak{T}_{nT}\, y(t)} \left(1 - q_{n,t}\right)^{1-\mathfrak{T}_{nT}\, y(t)} \tag{5.11}$$

Assuming the targeted data distribution $p_{\text{data}} \equiv p(\boldsymbol{y}, \boldsymbol{x})$ is empirically estimable from the given training set $p_{\text{data}} \simeq p(\boldsymbol{y}_{0:NT}, \boldsymbol{x}_{0:NT})$, then the objective is to minimize some sort of a measure of divergence between the true data distribution p_{data} (represented by its empirical estimate $p(\boldsymbol{y}_{0:NT}, \boldsymbol{x}_{0:NT})$) and the model's induced distribution $p_{\text{model}} = p(\boldsymbol{y},\, \boldsymbol{x} \,|\, \boldsymbol{\theta}_g)$ via adjusting parameters $\boldsymbol{\theta}_g$ of the latter. A standard choice is the $\underline{\text{K}}$ullback-$\underline{\text{L}}$eibler (KL) divergence denoted by $\mathrm{D}_{\mathrm{KL}}(\,\cdot \,||\, \cdot\,)$ and given for the two distributions p_{data} and p_{model} by

$$\mathrm{D}_{\mathrm{KL}}\left(p_{\text{data}} \;||\; p_{\text{model}}\right) = \mathrm{E}_{p_{\text{data}}}\left[\log\left(\frac{p_{\text{data}}}{p_{\text{model}}}\right)\right] \tag{5.12}$$

$$= \mathrm{E}_{p_{\text{data}}}\left[\log\ p_{\text{data}}\right] - \mathrm{E}_{p_{\text{data}}}\left[\log p_{\text{model}}\right] \tag{5.13}$$

$$= \mathrm{E}_{p_{\text{data}}}\left[\log\ p_{\text{data}}\right] - \mathrm{E}_{p_{\text{data}}}\left[\log p\left(\boldsymbol{y}, \boldsymbol{x} \mid \boldsymbol{\theta}_g\right)\right] \tag{5.14}$$

which is a function of the parameters $\boldsymbol{\theta}_g$ via its rightmost term. Accordingly, minimizing this divergence measure with respect to the model parameters $\boldsymbol{\theta}_g$ accounts to maximizing the expectation $\mathrm{E}_{p_{\text{data}}}[\log p(\boldsymbol{y}, \boldsymbol{x} \mid \boldsymbol{\theta}_g)]$ which is, in turn, equivalent to

$$\max_{\boldsymbol{\theta}_g} \mathrm{E}_{p_{\text{data}}}[\log p\left(\boldsymbol{y}, \boldsymbol{x} \mid \boldsymbol{\theta}_g\right)] = \mathrm{E}_{p_{\text{data}}}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_g)] + \mathrm{E}_{p_{\text{data}}}[\log p(\boldsymbol{x})]$$

$$= \mathrm{E}_{p_{\text{data}}}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_g)] + \text{const.} \tag{5.15}$$

where the general product rule $p\left(\boldsymbol{y}, \boldsymbol{x} \mid \boldsymbol{\theta}_g\right) = p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_g)p(\boldsymbol{x})$ has been utilized. Maximizing the log-likelihood function $\mathrm{E}_{p_{\text{data}}}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}_g)]$ based on samples from the joint distribution $p(\boldsymbol{y}_{0:NT}, \boldsymbol{x}_{0:NT})$ is a standard approach in machine learning known as maximum likelihood (ML) [Bishop 2006]. With the assumed Bernoulli distribution, the ML formulation is given finally by

$$\arg\max_{\boldsymbol{\theta}_g} \left(\sum_{n=0}^{N-1} \sum_{t=1}^{T} \mathfrak{T}_{nT} y(t) \log\left(q_{n,t}\right) + \left(1 - \mathfrak{T}_{nT} y(t)\right) \log\left(1 - q_{n,t}\right) \right) \tag{5.16}$$

and is our choice for estimating the parameters of the proposed model which is introduced next.

5.4.2. Model Architecture

As stated earlier, each distribution parameter $q_{t,n}$ is a function of the corresponding input sub-segment $\mathfrak{T}_{nT}\boldsymbol{x}_{0:T}$. We propose a deep convolutional neural net whose objective is to estimate a sub-segment $\boldsymbol{q}_n$ of the parameter sequence given by

$$\boldsymbol{q}_n = \left[q_{n,1}, q_{n,2}, \cdots, q_{n,T}\right]^{\mathsf{T}} \tag{5.17}$$

from the input sub-segment $\mathfrak{T}_{nT}\boldsymbol{x}_{0:T}$ such that

$$\boldsymbol{q}_n = g(\mathfrak{T}_{nT}\boldsymbol{x}_{0:T} \mid \boldsymbol{\theta}_g) \tag{5.18}$$

for any $n \geqslant 0$ where $\boldsymbol{\theta}_g$ denotes the model parameters (e.g. weights and biases of the neural net), $g : \mathbb{R}^T \to [0, 1]^T$ represents the neural network function (assuming $x(t) \in \mathbb{R}$).

In the following we detail the architecture of the neural network g that maps a given input sub-segment $\mathfrak{T}_{nT}\boldsymbol{x}_{0:T}$ to the corresponding sub-segment of the output distribution parameters $\boldsymbol{q}_n$. We first give a high level overview of the network architecture, and then detail each elementary transformation in its layers.

The proposed model is a fully convolutional neural network comprising 46 layers in five parts, namely, an input layer, 20 layers in pooling (aka downsampling) blocks, 4 intermediate layers, 20 layers in un-pooling (aka upsampling) blocks, and finally an output layer. The model size reaches around 41M of trainable parameters.

Figure 5.2 depicts the architecture of the proposed model. The model's input is a sub-segment of the aggregate signal $x(t)$ of length T denoted by $\mathfrak{T}_\tau\boldsymbol{x}_{0:T}$ for some $\tau \in \mathbb{Z}_{\geqslant 0}$ and the output is the predicted distribution parameters[10] for the same interval. The input segment length T can be varied dynamically (that is, during run-time) since the model consists merely of convolutional layers with optional pooling[11]. The segment length T must respect the pooling operations though, and has to be divisible by $3 \times 3 \times 3 \times 5 \times 5 = 675$ samples (that is roughly 11.25 minutes at 1 Hz sampling rate) for the proposed model under the existing implementation.

Computations proceed layer-wise where each layer is composed of elementary operations depicted in its middle section (e.g "`GN` ∘ `BN` ∘ `LReLU` ∘ `CONV`") and applied (from right to left) to the layer's input. For example, the first layer's output is given for the input sequence $\mathfrak{T}_\tau\boldsymbol{x}_{0:T}$ by

$$\left\{\texttt{GN} \circ \texttt{LogSg} \circ \texttt{BN} \circ \texttt{CONV}(7, 1)\right\}\left(\mathfrak{T}_\tau\boldsymbol{x}_{0:T}\right) \tag{5.19}$$

and each elementary operation (e.g. "`BN`") will be defined shortly.

[10]These parameters for the Bernoulli distributions are identical to mean predictions since $\mathbb{E}[x] = q$ whenever $x \sim$Bernoulli(q). This property will be utilized shortly.

[11]Convolution and pooling operations will be defined shortly (see Section 5.4.3). Additionally, the convenience (and advantage) of a dynamic segment length will be discussed in Section 5.4.6.

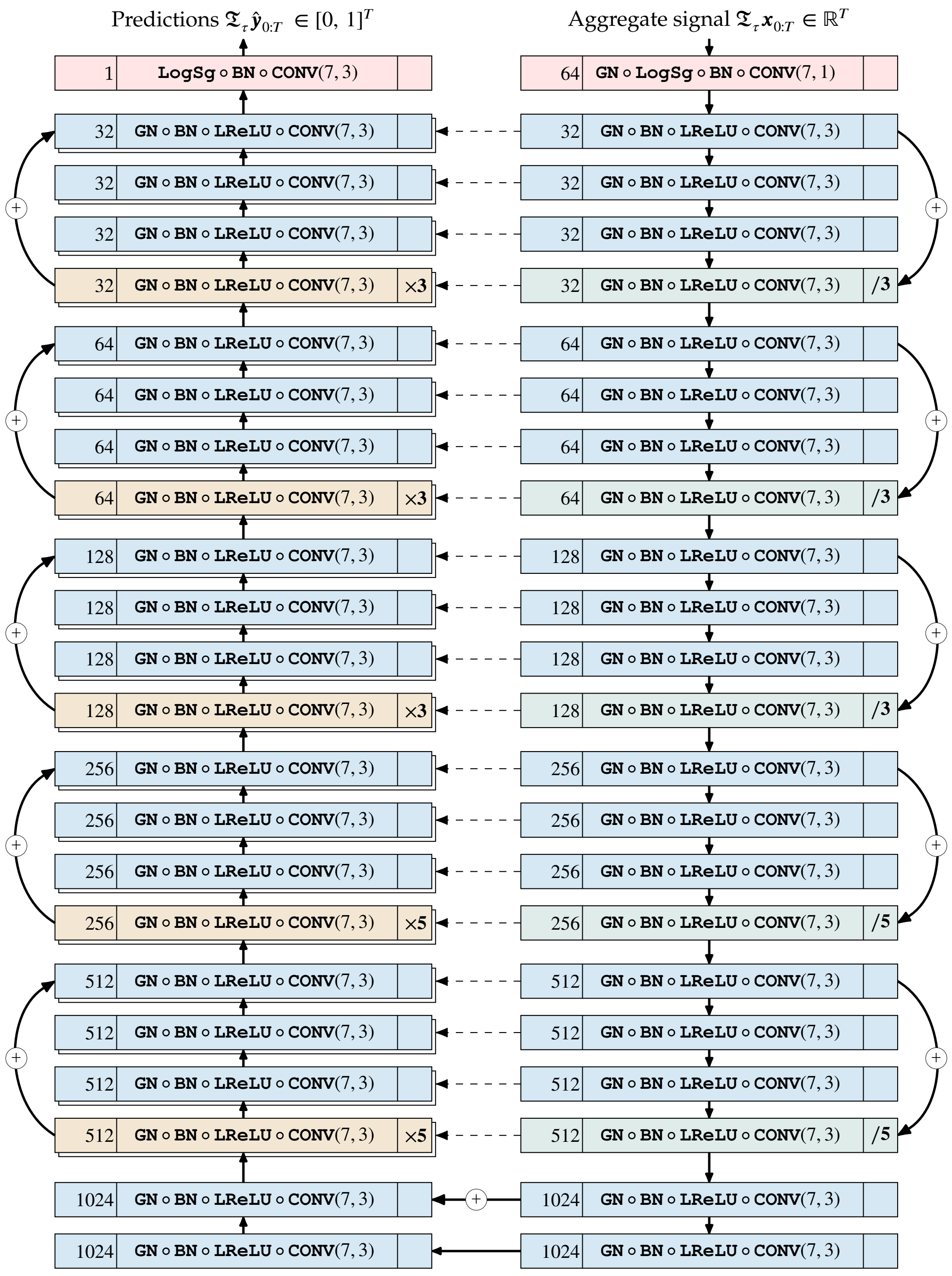

Predictions $\mathfrak{T}_\tau \hat{\boldsymbol{y}}_{0:T} \in [0,\ 1]^T$
Aggregate signal $\mathfrak{T}_\tau \boldsymbol{x}_{0:T} \in \mathbb{R}^T$
1 LogSg ∘ BN ∘ CONV(7, 3)
64 GN ∘ LogSg ∘ BN ∘ CONV(7, 1)
32 GN ∘ BN ∘ LReLU ∘ CONV(7, 3)
64 GN ∘ BN ∘ LReLU ∘ CONV(7, 3)
128 GN ∘ BN ∘ LReLU ∘ CONV(7, 3)
256 GN ∘ BN ∘ LReLU ∘ CONV(7, 3)
512 GN ∘ BN ∘ LReLU ∘ CONV(7, 3)
1024 GN ∘ BN ∘ LReLU ∘ CONV(7, 3)
×3
×5
/3
/5
+

Figure 5.2.: Architecture of the proposed model $\boldsymbol{g}(\,\cdot \mid \boldsymbol{\theta}_g\,)$ (description in text).

Each layer may optionally apply a pooling or un-pooling operation to its output or input, respectively, and this is characterized by the right section of each layer's block. Pooling and un-pooling factors are indicated by a divisor (e.g. "**`/3`**") for the former or a multiplier (e.g. "**`x3`**") for the latter. Additionally, pooling, un-pooling, input, and output layers are all highlighted with a distinct background colors.

The left section of each layer's block (e.g. 64) indicates the number of output channels c_{out} at that layer.

Standard solid lines represent the normal forward flow, while dashed lines represent skip connections with channel concatenation, and solid lines with addition nodes represent residual connections.

Skip-connection (aka. shortcuts or lateral connections) are direct connections from the low-level layers to the high-level ones through channel-wise concatenation. Such direct connections were empirically shown to deliver higher performance in both discriminative [Rasmus et al. 2015a;b, Ronneberger et al. 2015, Valpola 2015] or generative models [Isola et al. 2016]. The resulting architecture is usually referred to as U-Net [Ronneberger et al. 2015] or LadderNet [Rasmus et al. 2015b].

Residual connection are identity skip connections that comprise identity mappings across a block of layers (known as residual blocks) through element-wise addition to the target layer's output [He et al. 2016a;b, Chen et al. 2018]. Residual connections were proven to stabilize, simplify, and accelerate training hierarchical architectures with depth. We refer the interested reader to e.g. He et al. [2016b] for a detailed discussion on residual connections and their advantages in deep learning architectures. The resulting neural architecture is commonly referred to as ResNet [He et al. 2016a].

5.4.3. Elementary Operations

Elementary operations in model layers will be detailed in the sequel. For notational convenience, and without loss of generality, we shall drop dependence on layer channels, and consider merely the input sequence $\boldsymbol{x}_{0:T}$.

Dilated temporal convolutions $\texttt{CONV}(k, r)$: The core operation of each layer is the temporal convolution[12] defined as

$$\texttt{CONV}(k, r)\{\,\boldsymbol{x}_{0:T}\,\} \;=\; b + \mathbf{w} * \boldsymbol{x}_{0:T} \tag{5.20}$$

where r is the dilation rate [Yu and Koltun 2016], k is the kernel size, b is the bias term, $\mathbf{w}$ is the layer's kernel, and $*$ denotes the convolution operation defined appropriately for finite length sequences such that output remains a T dimensional sequence. Trainable parameters are the bias b and the kernel weights $\mathbf{w}$.

The original work by Yu and Koltun [2016] applied a geometrically increasing dilation rate with respect to layer depth whose outcome is an exponential expansion of the effective receptive field. We, however, restricted our models to a fixed dilation rate throughout all layers whose effect is merely a fixed scaling of the receptive field.

Kernel sizes are set to $k = 7$ while dilation takes place with a fixed rate of $r = 3$ for all layers except the input layer where no dilation takes place.

Batch normalization $\texttt{BN}$ [Ioffe and Szegedy 2015]: is a composition of two affine transformations applied to the output of each layer based on mini-batch statistics. The objective of batch normalization is to weakly constrain[13] the distribution of all layer inputs and, as a result, reduce what is known as covariate shifts.

Batch normalization is applied independently to each element (aka neuron) in a layer, and for the i^{th} element in the vector $\boldsymbol{x}_{0:T}$ is given by

$$\texttt{BN}\{x_i\} = \gamma\,\hat{x}_i + \beta = \gamma\,\frac{x_i - \mu_i^{(\mathscr{B})}}{\sigma_i^{(\mathscr{B})}} + \beta \tag{5.21}$$

where x_i is the i^{th} element of the vector $\boldsymbol{x}_{0:T}$, γ and β are two layer-wide trainable parameters, $\mu_i^{(\mathscr{B})}$ and $\sigma_i^{(\mathscr{B})}$ are the sample mean and standard-deviation of this element over the the training mini-batch $\mathscr{B} \subseteq \mathscr{D}$. In this work, we apply $\texttt{BN}$ prior to the non-linear activations as originally proposed in [Ioffe and Szegedy 2015] for the

[12]Since one of the convolution terms, namely the weights $\mathbf{w}$, is a trainable parameter, this deems the reflection in the definition of a convolutional operations unnecessary and the operation in practice takes the form of a cross-correlation.

[13]Through empirical normalization of the first two moments (aka zero-mean unit-variance standardization).

input and output layers only but found empirically that normalizing post the non-linear activation in the remaining layers showed better results. Moreover, we avoid deploying any dropout [Srivastava et al. 2014] regularization as proposed in [Ioffe and Szegedy 2015] and found in [Isola et al. 2016, Radford et al. 2016] to hurt the performance of a network that jointly includes both regularization techniques. Noise injection GN (introduced next) is in our case a valuable replacement for dropout.

Activation noise (aka Gaussian noise injection) GN [Nair and Hinton 2010]: activation noise is a regularization technique applied during the training phase only, and consists of injecting small additive white Gaussian noise to the layer's output to avoid over-fitting. This operation is defined simply by

$$\mathtt{GN}\{x(t)\} = x(t) + \omega(t) \tag{5.22}$$

where $\omega(t) \sim \mathcal{N}(0, \sigma^2_{\mathtt{GN}})$ for all t where the design hyper-parameter $\sigma^2_{\mathtt{GN}}$ is the variance of injected noise.

Leaky Rectified Linear Units LReLU [Maas et al. 2013]: is an activation function that represents the main source of non-linearity in the proposed model and is defined as

$$\mathtt{LReLU}\{x(t)\} = \max(\alpha x(t),\ x(t)) \tag{5.23}$$

where $\alpha < 1$ is a design hyper-parameter and is set in our model to $\alpha = 0.1$ for all layers, and $\max(a, b)$ returns the maximum amongst the two scalars a and b. Activation functions are applied element-wise to multivariate inputs.

Logistic sigmoidal activations LogSg: is a bounded activation function applied to the first hidden layer and the output layer of the model

$$\mathtt{LogSg}\{x(t)\} = \frac{1}{1 + \exp(-x(t))} \tag{5.24}$$

5.4.4. Training Procedure

In the proposed CNN architecture, trainable parameters $\boldsymbol{\theta}_g$ are the convolutional kernels $\mathbf{w}$ and biases b in addition to the batch normalization shift β and scale γ parameters in each layer. A distinct set of these parameters is optimized for each target load.

Initialization: biases b and shifts β are initialized with zeros across all layers while normalization scales γ are initialized with ones. Convolutional kernel weights $\mathbf{w}$ are initialized from a layer-dependent zero-mean uniform distribution denoted for the lth layer by $\mathcal{U}(-a^{(l)}, a^{(l)})$ whose parameter $a^{(l)}$ is given by

$$a^{(l)} = \sqrt{\frac{12}{k^{(l)}\left(c_{\text{in}}^{(l)} + c_{\text{out}}^{(l)}\right)}} \tag{5.25}$$

where $k^{(l)}$ is the kernel size, and $c_{\text{in}}^{(l)}$ and $c_{\text{out}}^{(l)}$ denote the number of input and output channels, respectively, for that layer [Glorot and Bengio 2010].

Optimization: the log-likelihood function Equation (5.16) is maximized with respect to all trainable parameters $\boldsymbol{\theta}_g$ using gradient-based optimization. Gradients are estimated from random mini-batches $\mathcal{B} \subseteq \mathcal{D}$ of the whole training set $\mathcal{D}$ in an approach commonly referred to as stochastic gradient optimization [Bottou 2012, LeCun et al. 1998a], and propagated to their corresponding parameters through automatic differentiation in reverse accumulation mode (that is, back-propagated) [Baydin et al. 2018]. The parameter update rule is based on the ADAM heuristics [Kingma and Ba 2014] with Nesterov momentums [Dozat 2015] whose details are outside the scope of this work. We exploit the same hyper-parameter values of both approaches as recommended in their original works.

Stopping: in order to avoid overfitting in such a highly parameterized model, optimization (aka data fitting) is performed as long as the model keeps improving on a held-out dataset, referred to as the validation set, and is terminated once further parameter updates start to hurt the performance on that dataset. Important, however, is that the validation set is never used for data fitting. This is approach commonly referred to as early stopping [Prechelt 2012].

5.4.5. Predictive Function

Upon deployment, the model receives a sub-sequence of the aggregate signal $\boldsymbol{x}_{0:T}$ to infer the mean predictions $\hat{\boldsymbol{q}} = g(\boldsymbol{x}_{0:T} \mid \hat{\boldsymbol{\theta}}_g)$ where $\hat{\boldsymbol{\theta}}_g$ is the estimated model parameters from the training process. The following threshold-based decision rule is then adopted to estimate the corresponding activation profiled

$$\hat{y}(t) = \begin{cases} 1 & \text{if } \hat{q}_t \geqslant \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \tag{5.26}$$

where $\hat{y}(t)$ is the estimated activation profile at time $0 < t \leqslant T$. The mapping is deterministic since we disable stochastic operations (e.g. noise injection) in test time.

5.4.6. Remarks and Deployment Considerations

First, we highlight that the concept of load activation cycles (that is, a complete cycle of operation resembling the state path *off* → *on* → *off* as for instance adopted by Kelly and Knottenbelt [2015a]), is not considered in our model. In other words, an activation cycle of a load can extend over several T-length windows (such as lighting circuits and dishwashers) or arise more than once within the same window (as in the case of fridge and kettle activations). This is an important property since a disaggregator need not wait till the deactivation of a load (i.e. switch-*off* event) but rather can provide near real-time feedback from a partial sub-segment of the aggregate signal. Such a property will be investigated in our experiments (see for instance Figure 5.4).

Pertinent yet is our second remark which concerns how close the sought feedback can be to real-time. We shall not discuss the computational requirements of the model since this is affected by numerous factors such as the target processing architecture, possibility of hardware acceleration or parallel computation, chances of model distillation, ... etc. Rather we shall discuss the need for future samples $x(t+\tau)$ with $\tau > 0$ for predicting the load activation $\hat{y}(t)$ at time t.

We first point out that the proposed model is a form of a fully convolutional neural net which in a sense behaves computationally as a single convolution in the sense that it attains a fixed size (that is, number of parameters) but applies to any segment length T as long is it respects the involved pooling[14]. Additionally, we point out that

[14]As fractional pooling was not supported.

the two-sided[15] receptive field[16] of a neuron in the output layer is around 17 hours at a sampling frequency of 1 Hz. In other words, no benefit will be obtained from training on sub-sequences of length $T > 17$ hours, and in our experiments we used only 3-hour segments. Accounting for downsampling[17], a given sequence length T must be divisible by 675 which, in turn, demands a lower bound on the sequence length $T \geqslant 11.25$ minutes.

Due to the use of a fully convolutional architecture, dimensionality of model parameters are independent of the length of the input sequence T. Kernel dimensionality is given by its size and the number of input and output channels, while biases b, shifts β, and scales γ are defined channel-wise. Hence, the sequence length T can be varied dynamically, and that corresponds to how much contextual information is provided to the model for a given query point. In other words, while T remains a design parameter that helps the model train on contextual information, the model can be trained using adequate length T but the user may decide in the deployment phase to trade accuracy for computational efficiency[18] and query the model with a much shorter sequence. This renders the proposed model adjustable for performance, delay, and computational cost trade-offs.

Second, we bring to discussion causal requirements of the proposed model. As stated earlier, the model predicts the load activation $\hat{y}(t)$ at time t where $T < t \leqslant 2T$ from the corresponding sub-sequence $\mathfrak{T}_T \boldsymbol{x}_{0:T}$. This implies a causal inference for $\hat{y}(2T)$ and an anticausal[19] inference for $\hat{y}(T+1)$. But the choice of the reference point is arbitrary, and $\hat{y}(T+1)$ can be similarly causally inferred from past samples $\mathfrak{T}_1 \boldsymbol{x}_{0:T}$. In fact, a prediction $\hat{y}(t)$ at time t can be inferred from $\mathfrak{T}_\tau x_{0:T}$ where $t - T \leqslant \tau < t$. Inference is causal if $\tau = t - T$, anticausal if $\tau = t - 1$, and acausal in between. This directly raises the question of prediction performance based on contextual information. In all our experiments, we have not observed noteworthy variations in

[15]Considering both historical and future samples. Half that size is inferred from the past and present samples, while other half considers future samples.

[16]We define the receptive field of a neuron as the total number of input layer nodes (that is, maximum-length input sub-segment) that *can* be used to determine the value of that neuron. This is irrelevant to the actual input length T since it can be varied dynamically.

[17]That is, the product of all downsampling factors (3, 3, 3, 5, 5).

[18]A fully convolutional neural net acts computation-wise as a single convolution operation $\boldsymbol{x}(t) * \mathbf{w}$. Such an operation has a fixed size (that is, the kernel width $\mathbf{w}$) but scales in the number of operations with the width of the input signal $\boldsymbol{x}(t)$.

[19]We define an anticausal system as one that does not depend on historical inputs, causal system as one that does not depend on future inputs, and acausal system if depends on both past and future inputs.

performance for a 3-hour monitoring period[20]. We adopted a non-overlapping window for inference where the very first and last samples are anticausal and causally inferred, respectively, while everything else remains acausal. This is of course poses a delay (a 3-hour delay) if the proposed model is adopted for real-time monitoring and disaggregation. However, since the disaggregation performance across all output neurons (regardless of *causal-ness*) is indistinguishable, one can easily realize real-time disaggregation with the proposed model at a computation cost. We encourage future investigation of the effect of this feature on the disaggregation performance and the cost of seeking real-time monitoring.

5.5. Time Series Performance Assessment

Early works on energy disaggregation favored the conventional accuracy index Equation (5.27) in evaluating the performance of an energy disaggregation model with respect to activation profiles in single-load extraction frameworks [Chang et al. 2010, Belkin 2013, Makonin et al. 2013]. Later works, however, realized the misleading interpretation of this biased measure and alternatively proposed the measures precision and recall in addition to their harmonic mean F_1-score, commonly adopted in the domain of information retrieval, for assessing disaggregation performance [Christian et al. 2014, Holmegaard and Kjaergaard 2016, Kim et al. 2010, Makonin and Popowich 2015]. The misleading interpretation of the accuracy index resulted from its bias toward the prevailing class. Such a biased measure falsely credits a trivial solution that results from constantly predicting the prevailing class with a high score value [Makonin and Popowich 2015]. In the context of domestic energy disaggregation, for example, scarcity of load usage leads to high prevalence of the negative class (i.e. the *off* state). This, in turn, leads to a high accuracy value assigned to a trivial solution that consistently predicts the load to be *off* regardless of the input signal.

[20]This observation is most likely due to the fact that each output neuron spans the whole 3-hour input segment with its receptive field. Performance variations might be revealed if 1) the input segment is longer the *half* the receptive field of the output layer (that is, longer than $\sim$ 8.5 hours) and 2) contextual information is valuable for the target monitoring beyond that range. In that case, Some middle output neurons will leverage more contextual information than e.g. the right-most causal neuron.

We, however, believe that these measures of precision and recall represent a one-sided rigorous solution to the biasedness property of the Rand accuracy[21]. In fact, these metrics are fused to the assumption of scarce load usage and may fail to provide valuable interpretation of performance whenever this assumption is violated (e.g. in commercial and industrial settings or for repetitively activated and long-running loads). In the following, we briefly illustrate the problem with all previous measures and propose alternative, more balanced and flexible solutions, namely, Markedness and Informedness as proposed by Powers [2011], that will be adopted in the rest of this work. For this purpose, we shall utilize the notation adopted by Powers [2011] and define the two contingency tables as

		Predictions		
		$\hat{y}^+$	$\hat{y}^-$	
Classes	y^+	TP	FN	RP
	y^-	FP	TN	RN
		PP	PN	N

TP:	True positives
TN:	True negatives
FP:	False positive
FN:	False negatives
RP:	Real positives
RN:	Real negatives
PP:	Positive predictions
PN:	Negative predictions
N:	Number of samples/events

and

		Predictions		
		$\hat{y}^+$	$\hat{y}^-$	
Classes	y^+	tp	fn	rp
	y^-	fp	tn	rn
		pp	pn	1

tp:	Ratio of true positives $p(y^+, \hat{y}^+)$
tn:	Ratio of true negatives $p(y^-, \hat{y}^-)$
fp:	Ratio of false positives $p(y^-, \hat{y}^+)$
fn:	Ratio of false negatives $p(y^+, \hat{y}^-)$
rp:	Ratio of the positive class $p(y^+)$
rn:	Ratio of the negative class $p(y^-)$
pp:	Ratio of positive predictions $p(\hat{y}^+)$
pn:	Ratio of negative predictions $p(\hat{y}^-)$

where y^+, y^- represent the true positive and negative class assignments, and similarly $\hat{y}^+$, $\hat{y}^-$ for assignments of the estimated class. Note that TP, TN, FP, FN, RP, RN, PP, PN, and N are raw counts while tp=TP/N, tn=TN/N, fp=FP/N, fn=FN/N,

[21]Hereafter, we shall use the term Rand accuracy to refer to the biased accuracy measure as it is similar to the Rand index [Rand 1971] adopted in data clustering in the case of binary clustering and to distinguish this measure from the unbiased accuracy introduced later.

$\mathtt{pp}=\mathtt{PP}/\mathtt{N}$, $\mathtt{pn}=\mathtt{PN}/\mathtt{N}$, $\mathtt{rp}=\mathtt{RP}/\mathtt{N}$, and $\mathtt{rn}=\mathtt{RN}/\mathtt{N}$ represent empirical estimates of the joint probabilities $p(y^+,\hat{y}^+)$, $p(y^-,\hat{y}^-)$, $p(y^-,\hat{y}^+)$, $p(y^+,\hat{y}^-)$, the marginals $p(\hat{y}^+)$, $p(\hat{y}^-)$, and class priors $p(y^+)$, $p(y^-)$, respectively, (for $\mathtt{N} \to \infty$)[22]. Rand accuracy $\mathtt{acc}$ is then defined as

$$\mathtt{acc} = \frac{\mathtt{TN + TP}}{\mathtt{TN + TP + FN + FP}} \tag{5.27}$$

$$= \mathtt{rn} \cdot \mathtt{TNR} + \mathtt{rp} \cdot \mathtt{TPR} \tag{5.28}$$

$$= \mathtt{tn} + \mathtt{tp} \tag{5.29}$$

where

$$\mathtt{TPR} = \frac{\mathtt{TP}}{\mathtt{TP + FN}} = \frac{\mathtt{TP}}{\mathtt{RP}} = \text{recall} \tag{5.30}$$

is the true positive rate or *recall* (the ratio of correctly retrieving positives w.r.t all real positives) and

$$\mathtt{TNR} = \frac{\mathtt{TN}}{\mathtt{TN + FP}} = \frac{\mathtt{TN}}{\mathtt{RN}} = \text{inverse-recall} \tag{5.31}$$

is the true negative rate or *inverse recall* (ratio of correctly retrieving negative events w.r.t real negatives). Rand accuracy can be seen as a weighted average of the recall and it inverse eq. (5.28). The weights in eq. (5.28) are the empirical estimates of the class priors. As a result, Rand accuracy will be biased toward the prevailing class (the class of the dominant prior).

In the aforementioned situation of scarce load usage, the probability of negative samples becomes relatively high ($\mathtt{rn} \to 1$) and Rand accuracy becomes a single-sided measure, namely the inverse recall ($\mathtt{acc} \to \mathtt{TNR}$), involving no trade-off between retrieval and relevance. Maximizing the inverse recall $\mathtt{TNR}$ in this case (e.g. via a trivial solution retrieving all negative events) ambiguously yields near optimal values of the Rand accuracy measure, while correctly retrieving positive events will have minor improvements.

The information retrieval approach to alleviate this bias is to simply ignore the prevailing term in eq. (5.27), namely $\mathtt{TN}$. This approach either results in the Jaccard

[22]Here $p(y^+)$ is in fact $p(y = y^+)$ or $p(y = 1)$, and similarly for all other terms, but we avoid this pedantry for simplicity unless it becomes a source of confusion.

index [Levandowsky and Winter 1971] or the F_1-score ($\mathtt{F1S}$) defined as

$$\mathtt{F1S} = \frac{\mathtt{TP + TP}}{\mathtt{TP + TP + FN + FP}} \tag{5.32}$$

$$= \frac{2 \cdot \mathtt{TPR} \cdot \mathtt{TPA}}{\mathtt{TPR} + \mathtt{TPA}} \tag{5.33}$$

where

$$\mathtt{TPA} = \frac{\mathtt{TP}}{\mathtt{TP + FP}} = \text{precision} \tag{5.34}$$

is the true positive accuracy or *precision*. The F_1-score is the harmonic mean of precision and recall and both are measures that simply ignore the correct prediction of the negative class. Nevertheless, precision $\mathtt{TPA}$ and recall $\mathtt{TPR}$ remain biased measures. The difference between these measures and the Rand accuracy index is that the latter is biased to the prevailing class, while precision and recall are statically biased to the positive class.

Intuitively, a more robust solution to the problem of class imbalance is to readjust the weighted sum in the Rand accuracy measure so as to account for variations in class distributions

$$\text{unbiased accuracy} = \frac{\mathtt{TN} \cdot \mathtt{rp} + \mathtt{TP} \cdot \mathtt{rn}}{\mathtt{TN} \cdot \mathtt{rp} + \mathtt{TP} \cdot \mathtt{rn} + \mathtt{FP} \cdot \mathtt{rp} + \mathtt{FN} \cdot \mathtt{rn}} \tag{5.35}$$

$$= \frac{\mathtt{TNR + TPR}}{2} \tag{5.36}$$

$$= \frac{1}{2}\left(\frac{\mathtt{tn}}{\mathtt{rn}} + \frac{\mathtt{tp}}{\mathtt{rp}}\right) \qquad \in [0,1] \tag{5.37}$$

In fact, we shall observe in due course that precision and recall are the result of extreme weighting where the positive and negative predictions are assigned the weights one and zero, respectively.

The question rises, however, as to why these highly biased measure are commonly adopted and widely accepted in the domain of information retrieval. Our claim is that, in that application domain $\mathtt{TN}$ is undefined or unknown rather than highly valued. The result is, therefore, an incomplete contingency matrix from which some aggregate measures can not be obtained. In the other words, it is assumed that the size of the test set is very large $\mathtt{N} \rightarrow \infty$ compared to the number of retrieved documents, and the number of irrelevant documents is usually unknown or difficult to estimate. As TN approaches N, it actually approaches infinity and weighting by TN will result in the (1, 0) weights and, in turn, to the precision and recall.

In energy disaggregation, and with the scarce usage assumption, TN is relatively high but known and upper bounded by N. Therefore, we believe that the (1, 0) weighting is an extreme ill-argued choice. Moreover, we argue that the scarce usage assumption is not necessarily valid for all load categories. For example, the fridge, air conditioner, space heaters, electric vehicles ... etc, are all domestic loads that often have an approximately balanced class distribution over their time of use. In these cases, Rand accuracy is a well interpreted measure. This is also the case for commercial and industrial loads [Batra et al. 2014b]. Additionally, loads that are *on* most of the time (which is a well known category of loads) would suffer from the opposite situation in which the prevailing class is the positive one. In this case, a trivial classifier that always predicts the positive class (i.e. the load is always *on*) will have a misleading high F_1-score value since TP becomes relatively high. Furthermore, when the scarce usage assumption is valid (e.g. for various miscellaneous appliances such as kettles, irons, vacuum cleaners ... etc), the extent of class imbalance varies widely amongst loads as well as users. These variations are not reflected by any means in either of the information retrieval measures. For these reasons, we claimed that precision and recall are inflexible measures since they are fused to a given assumption regardless of the real distribution of classes.

Powers [2011] introduced *informedness* B, *markedness* M, and their geometric mean *Matthews Correlation Coefficient* MCC Matthews [1975] as alternative, unbiased evaluation measures defined as

$$\mathtt{B} = \mathtt{TPR} + \mathtt{TNR} - 1 \tag{5.38}$$

$$\mathtt{M} = \mathtt{TPA} + \mathtt{TNA} - 1 \tag{5.39}$$

$$\mathtt{MCC} = \pm\sqrt{\mathtt{B}\cdot\mathtt{M}} \tag{5.40}$$

$$= \frac{\mathtt{TP}\cdot\mathtt{TN} - \mathtt{FP}\cdot\mathtt{FN}}{\sqrt{(\mathtt{TP}+\mathtt{FP})\cdot(\mathtt{TP}+\mathtt{FN})\cdot(\mathtt{TN}+\mathtt{FP})\cdot(\mathtt{TN}+\mathtt{FN})}} \quad \in [-1, 1] \tag{5.41}$$

where

$$\mathtt{TNA} = \frac{\mathtt{TN}}{\mathtt{TN} + \mathtt{FN}} = \text{inverse-precision} \tag{5.42}$$

is the true negative accuracy or *inverse-precision*. In fact, it is easily observed that informedness is simply a shifted and rescaled version of the balanced accuracy to the range $[-1, 1]$

$$\texttt{B} = 2 \times \text{unbiased-accuracy} - 1$$

The magnitude of the correlation MCC quantifies the level of agreement/disagreement between the model and the ground truth where $|\texttt{MCC}|=1$ indicates perfect agreement/disagreement versus chance $\texttt{MCC} \to 0$.

Similar to the information retrieval measures, these alternatives were also proposed and adopted in similar application domains. In medical diagnostics [Youden 1950, Metz 1978], for example, skew-insensitive measures of diagnostic performance are required for a comparative study between different tests over varying populations (e.g. test groups)[23]. Similarly, informedness B, markedness M, and MCC are utilized in recommender system evaluations [Schröder et al. 2011]. We believe that the requirements of performance evaluation in these applications are more similar to those in energy disaggregation.

These biased metrics of precision and recall can be viewed as one extreme of the more balanced metrics of informedness and markedness where the other extreme is their inverses. In fact, it is easily shown that both *informedness* and *markedness* converge to the traditional measures of precision and recall (or their inverses in the contrary case) as the misleading situation of highly skewed highly biased test is approached. For example, the common scenario in energy disaggregation where the negative class is prevalent, a trivial disaggregator which always predicts the negative class (resulting in a relatively high number of true negatives) would be evaluated based on the precision and recall measures

$$\lim_{\texttt{TN}\to\infty} \left(\frac{\texttt{TP}}{\texttt{TP}+\texttt{FN}} + \frac{\texttt{TN}}{\texttt{TN}+\texttt{FP}} - 1 \right) = \frac{\texttt{TP}}{\texttt{TP}+\texttt{FN}} = \text{recall}$$
$$\lim_{\texttt{TN}\to\infty} \left(\frac{\texttt{TP}}{\texttt{TP}+\texttt{FP}} + \frac{\texttt{TN}}{\texttt{TN}+\texttt{FN}} - 1 \right) = \frac{\texttt{TP}}{\texttt{TP}+\texttt{FP}} = \text{precission}$$

while the other extreme case will be evaluated based on the inverse precision and recall and all other scenarios will have a weighted average. For these reasons, we use these unbiased measures of informedness, markedness, and balanced accuracy in evaluating our models. Additionally, we provide both F_1–score and negative class prevalence rn for comparison with other works.

[23]In medical diagnostics, informedness is known as Youden' J-statistic [Youden 1950] and in human psychology markedness is known as DeltaP [Powers 2011].

5.6. Experiments and Results

The proposed model is empirically evaluated on one of the freely available energy monitoring datasets that is developed specifically for residential load disaggregation, namely UK-DALE [Kelly and Knottenbelt 2015a]. This section briefly summarizes relevant aspects of the UK-DALE dataset and the adopted evaluation setup in Section 5.6.1.

Afterwards, Section 5.6.2 assesses the feasibility of the proposed model and compares against relevant prior art. Finally, we report new benchmarking results for a long-term deployment in Section 5.6.3.

5.6.1. Training and Evaluation Dataset

The proposed model is evaluated on the publicly available UK-DALE dataset [Kelly and Knottenbelt 2015b], an energy dataset acquired from five residential buildings in the UK. In our experiments, we leverage the post-computed 1 Hz whole-house real power measurements while the target signals are the 0.1667 Hz real power measurements up-sampled to 1 Hz using forward filling. Conventionally, measurements are split into three folds for training, validation, and evaluation with more details provided in each of the following experimental setup (that is, Section 5.6.2 and Section 5.6.3). The input to each model is always a 3-hour window that is sliding in a non-overlapping scheme over the 24-hour day, and that is independent of the target load. Worth noting is that, training was merely performed on real measurements. In other words, in spit of the ease of semi-synthetic data generation for energy disaggregation, none of the following experiments utilized this advantage for data augmentation. Additionally, no extraction of activation cycles (that is, start and end of a single operation of a load) was necessary for the proposed model. Under these specifications, we consider two evaluation setups.

We refer the interested reader to the original work of Kelly and Knottenbelt [2015b] for a through description of the UK-DALE dataset.

Table 5.2.: Summary statistics of some of the selected loads from the first building in UK-DALE within the period from Jan. 2014 to Dec. 2016 [Kelly and Knottenbelt 2015b]. Loads are a kettle (KT), a microwave oven (MC), a dishwasher (DW), a washing machine (WM), a lighting circuit (LC), and a fridge (FR). The percent-noisy measure % – NM is the one proposed in [Makonin and Popowich 2015] and defined as ratio of the remaining residuals upon excluding the target load $\sum_t |\mathtt{P}(t) - \mathtt{P}^{(m)}(t)|$ to the aggregate signal $\sum_t \mathtt{P}(t)$. The total energy consumed is for that period is roughly 3,391 kW·H with the nominal power consumption being approximating 7.12 kW.

Load	KT	MC	DW	WM	LC	FR
Power [kW]	2.45	1.63	2.43	2.2	0.362	0.41
Energy [kW·H]	134	57.3	235	238	265	351
Energy share	3.98%	1.69%	7.02%	7.04%	7.81%	10.35%
Percent noisy	96.28%	97.99%	93.15%	93.06%	92.19%	89.74%
Usage time [day]	1.5	2.0	11.4	20.4	134.9	153.0
Percent active	0.41%	0.56%	3.11%	5.59%	36.91%	42.74%
# Activations	2,290	2,687	463	308	4283	8,887

5.6.2. Prior Art Validation

The first experimental setup targets major loads in the first building of UK-DALE. These include a kettle (KT), a microwave oven (MC), a dishwasher (DW), a washing machine (WM), a lighting circuit (LC), and a fridge (FR).

Table 5.2 shows several annual statistics of these loads estimated from the target building in a three-year duration from January 2014 to December 2016. The table shows that the set of selected loads represent around one third of the total energy consumption in the building out of 52 sub-metered channels in that building.

In this experiment, we compare the proposed model to prior art on the same dataset, namely the time-of-use regressor proposed by Kelly and Knottenbelt [2015a]. We adopt identical evaluation framework as in Kelly and Knottenbelt [2015a] which we briefly describe here for the sake of completeness.

Kelly and Knottenbelt [2015a] considered two experimental setups. The first is referred to as the *same-instance* setup, where a model is trained on a target load instance in a building, and is expected to monitor the same load instance in future profiles. In the second setup, the aim is to assess model generalization *across buildings*. Towards that end, a model is trained on multiple instances of the same load category across different buildings, and is expected to generalize to the same load category in new buildings.

Table 5.3.: Performance comparison between our proposed model and the *rectangles* architecture in [Kelly and Knottenbelt 2015a] under the same load instances (left) and unseen instances from new buildings (right). All values represent the F_1-score measure. Training and evaluation folds are adopted from [Kelly and Knottenbelt 2015a] where each model is trained on the whole data for the target load (that is between 2013 and 2015 at the time of their work) in the first house except for the very last week that is reserved for the *same instances* evaluation. The *across buildings* evaluation utilized a predefined set of training and test buildings for each load, and this is found in Table 3 in the aforementioned work.

Load	same instances		across buildings	
	2015a	ours	2015a	ours
FR	0.810	**0.879**	0.820	**0.927**
DW	0.720	**0.796**	0.740	**0.804**
MC	0.620	**0.705**	0.210	**0.366**
WM	0.490	**0.960**	0.270	**0.410**
KT	0.710	**0.783**	0.700	**0.819**

In Table 5.3, we compare the monitoring performance of our model with the previous work in [Kelly and Knottenbelt 2015a], specifically the regression-based model (referred to as the *rectangles* architecture in the aforementioned work). The table shows both evaluation setups proposed in the original work. For a fair comparison, we adopt identical data folds[24] as utilized by Kelly and Knottenbelt [2015a] for both evaluation setups and for each load. We, however, train only on real measurements while the original work additionally utilized synthetic data[25]. In the *same-instance* setup, the model trains on around 20-months of measurements from the first building of UK-DALE and is evaluated on the following one week. In the *across-buildings*

[24]We refer the interested reader to the original work by Kelly and Knottenbelt [2015a] (Tables 1-3) for a detailed description.

[25]Kelly and Knottenbelt [2015a] synthesized disaggregation setups by simply aggregating individual load profiles from a random subset of loads. Training data is then enriched with these synthesized profiles along with the selected loads as ground truth.

setup, the model trains on all measurements from a selected subset of buildings, and is evaluated on a new, distinct building.

As depicted in the Table 5.3, the proposed model consistently outperforms prior art on the same dataset in both evaluation setups and across all load categories. We report here merely the biased F_1-score for the sake of comparison, while the proposed assessment measures will be reported for our new benchmarking experiments in Section 5.6.3.

A notable observation yet is the poor generalization performance of two loads, namely the microwave `MC` and the washing machine `WM`, to new buildings when compared to other major loads e.g. the fridge `FR` or the kettle `KT`. A possible explanation from our point of view is the variations in loads' signatures across buildings for each end-use load category. For example, Parson et al. [2015] illustrates that washing machines show exceptional variation in their energy consumption across over 600 households compared to e.g. a dishwasher, a fridge, or even lights. This is consistent with our findings that washing machines disaggregators poorly generalize from one building to another. Admittedly, variations in loads' signature is not the sole cause of variations in the total energy consumption in a load category (e.g. variation in the consumption can be attributed to user behavior as well). However, for some loads that are likely to share the same level of behavioral variations (e.g. a dishwasher and a washing machine which is also verifiable from Table 5.2) we can explain away variations in the energy consumption via shifts in loads' signatures.

5.6.3. Long-term Validation

In this setup, we evaluate the proposed model on a selected set of loads from the first building in UK-DALE. Targeted loads includes major energy loads (fridge, washing machine, dishwasher, kettle, microwave oven, and lighting) in addition to few more loads, namely, solar thermal pump (`SP`), a television (`TV`), a boiler (`BL`), a toaster (`TS`), and the kitchen lamp (`KL`). For each of these loads, we train a distinct model but all share the same architecture, hyperparameters, and training procedure. Data folds are real power measurements from January and February of 2014 for training and validation, respectively, while the remaining 10 months of the 2014 represents the evaluation fold.

Table 5.4.: Long-term performance assessment of the model proposed in Figure 5.2 on a extended variety of loads from the first building in [Kelly and Knottenbelt 2015b] where training is performed on the first two months of 2014, and evaluation is reported on the remaining 10 months of the same year. Reported also are the proposed measures of informedness B, markedness M, and Matthews Correlation Coefficient MCC in addition to the probability of the negative class in the evaluation fold rn and the *percent-noisy measure* %-NM [Makonin and Popowich 2015]. Loads are a fridge (FR), lighting circuit (LC), a dishwasher (DW), a washing machine (WM), a solar thermal pump (SP), a television (TV), a boiler (BL), a kettle (KT), a microwave oven (MC), a toaster (TS), and kitchen lights (KL).

	rn	%-NM	TPA	TPR	B	M	F1S	MCC
FR	0.55	0.87	0.92	0.88	0.81	0.82	**0.896**	**0.815**
LC	0.69	0.92	0.52	0.67	0.39	0.35	**0.589**	**0.373**
DW	0.98	0.95	0.85	0.49	0.49	0.84	**0.623**	**0.641**
WM	0.94	0.91	0.97	0.99	0.99	0.96	**0.979**	**0.978**
SP	0.78	0.97	0.46	0.24	0.16	0.27	**0.312**	**0.204**
TV	0.90	0.97	0.74	0.69	0.67	0.71	**0.716**	**0.686**
BL	0.91	0.95	0.34	0.75	0.60	0.31	**0.468**	**0.431**
KT	0.99	0.95	0.87	0.87	0.87	0.87	**0.870**	**0.869**
MC	0.99	0.97	0.62	0.46	0.45	0.62	**0.526**	**0.529**
TS	0.99	0.98	0.67	0.72	0.72	0.67	**0.698**	**0.697**
KL	0.87	0.94	0.46	0.55	0.46	0.39	**0.502**	**0.422**

Table 5.4 shows the performance measures of the proposed model evaluated on the targeted 11 loads from the first building in the UK-DALE dataset with a 3-hour monitoring window for all load categories. We provide these results as an assessment of the feasibility of our proposed model and benchmarks for the addressed dataset. We additionally report the proposed performance measures of informedness B, markedness M, and Matthews correlation coefficient MCC.

It is observed how MCC matches the F_1-score in some loads, but is more conservative in most cases. In fact, biasedness of the F_1-score can be easily observed in loads showing the opposite of the scarce load usage assumption, primarily lights as in LC and KL, where it clearly deviates from MCC and provides an optimistic estimate of performance.

For an illustration, imagine switching the positive class of the lighting circuit (LC) so that it indicates that lights are switched-off. For the problem at hand, this sounds like a design choice, but in fact sends a message to the F_1-score measure that lights-off is a more important class to retrieve, and since it is fused to the positive class its values will notably degrade from the ones given in Table 5.4. On the other hand, `MCC` remains neutral to the choice of the positive and negative classes.

Figures 5.3 and 5.4 depict a 9-day sample of the disaggregation performance from Saturday, Dec. the 13th to 21st of 2014. Figure 5.3 show the aggregate real power profile (red curve) with the operating intervals of the washing machine highlighted in shaded areas (green shading). The operation periods are estimated from the sub-metered signals of this load (depicted in Figure 5.4 as the green curve) using the activation profile estimation technique detailed in Section 5.3.2.

Figure 5.4 shows the sub-metered real power signal of the washing machine (green curve) along with the estimated activation profile (red shading) using the proposed model. The figure not only illustrates the accuracy of the estimated activation profile, but additionally illustrates one of the key aspects of the proposed model. As illustrated, the model does not need a complete *on-off* operation period for estimation. Knowing that the trained model utilizes non-overlapping 3-hour windows (aligned with midnight) for prediction, one observes that some 3-hour windows comprise no operation period, some comprise a single period, and some even more. In all cases, the model is able to equally predict the whole activation profile regardless of the number of operation periods. In other cases, a single operation extends over two disaggregation windows (e.g. from Saturday Dec. the 13th to following day) with model reliably extracting the targeted activation profile.

Figure 5.5, additionally, illustrates a promising extension of the proposed model. In this experiment, we tweaked the proposed model to estimate the real load profile (rather than its activation profile) by replacing the cross-entropy loss function in Equation (5.16) with a $\underline{m}$ean-$\underline{s}$quared-$\underline{e}$rror (MSE) and leveraging the sub-metered real power profile of the load rather than the estimation detailed in Section 5.3.2. Since this approach is not thoroughly investigated in this work, we briefly show how promising this approach is through the visual illustration of the estimated load profile depicted in Figure 5.5 as a motivation of future work.

As depicted in the figure, the model was able to retrieve all operation intervals precisely. Observable also is that, the reconstructed load profile is notably smoother than the true sub-metered signal.

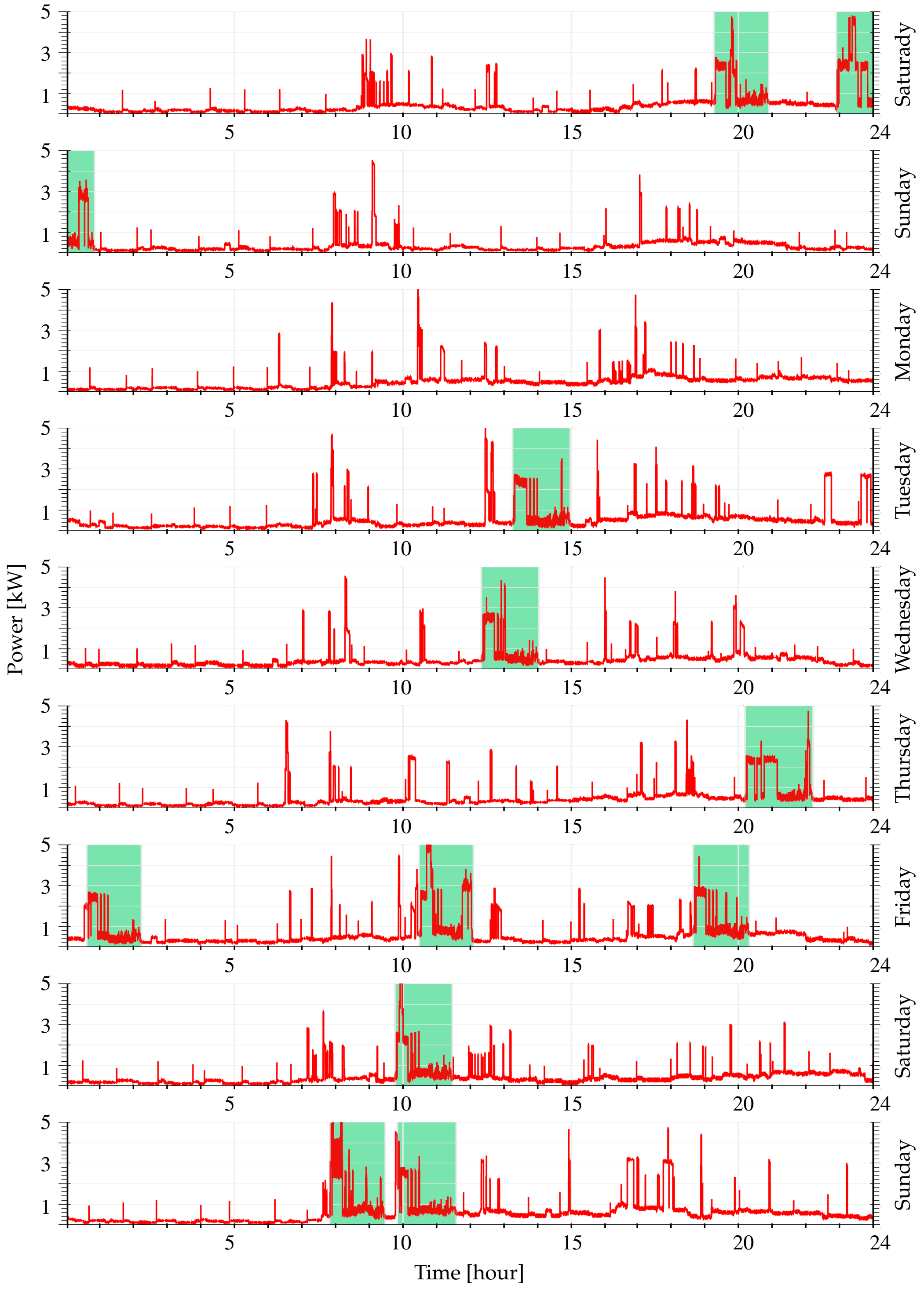


Figure 5.3.: Sample aggregate real power of UK-DALE (description in text).

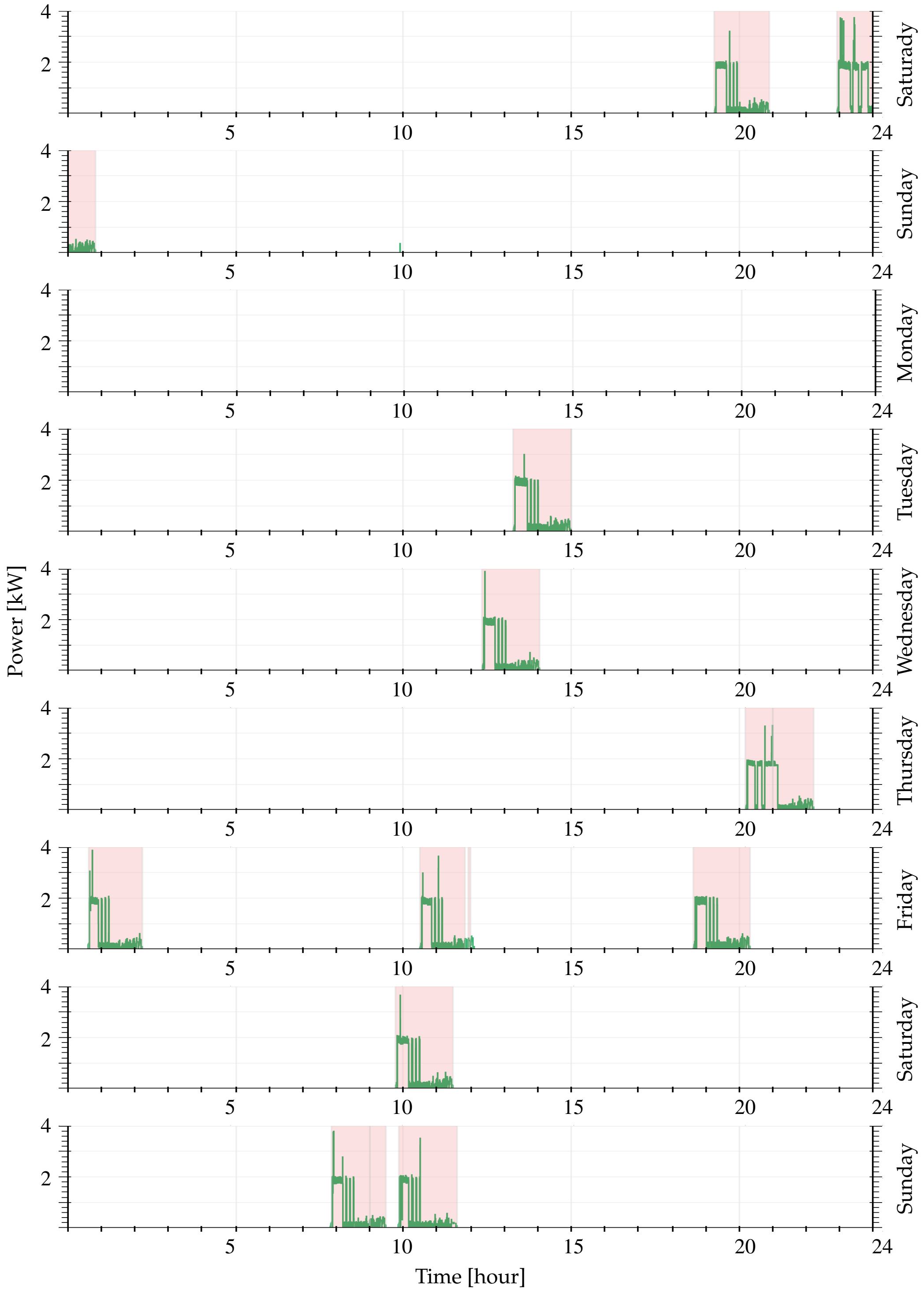


Figure 5.4.: `WM` real power and activation profile (details in text).

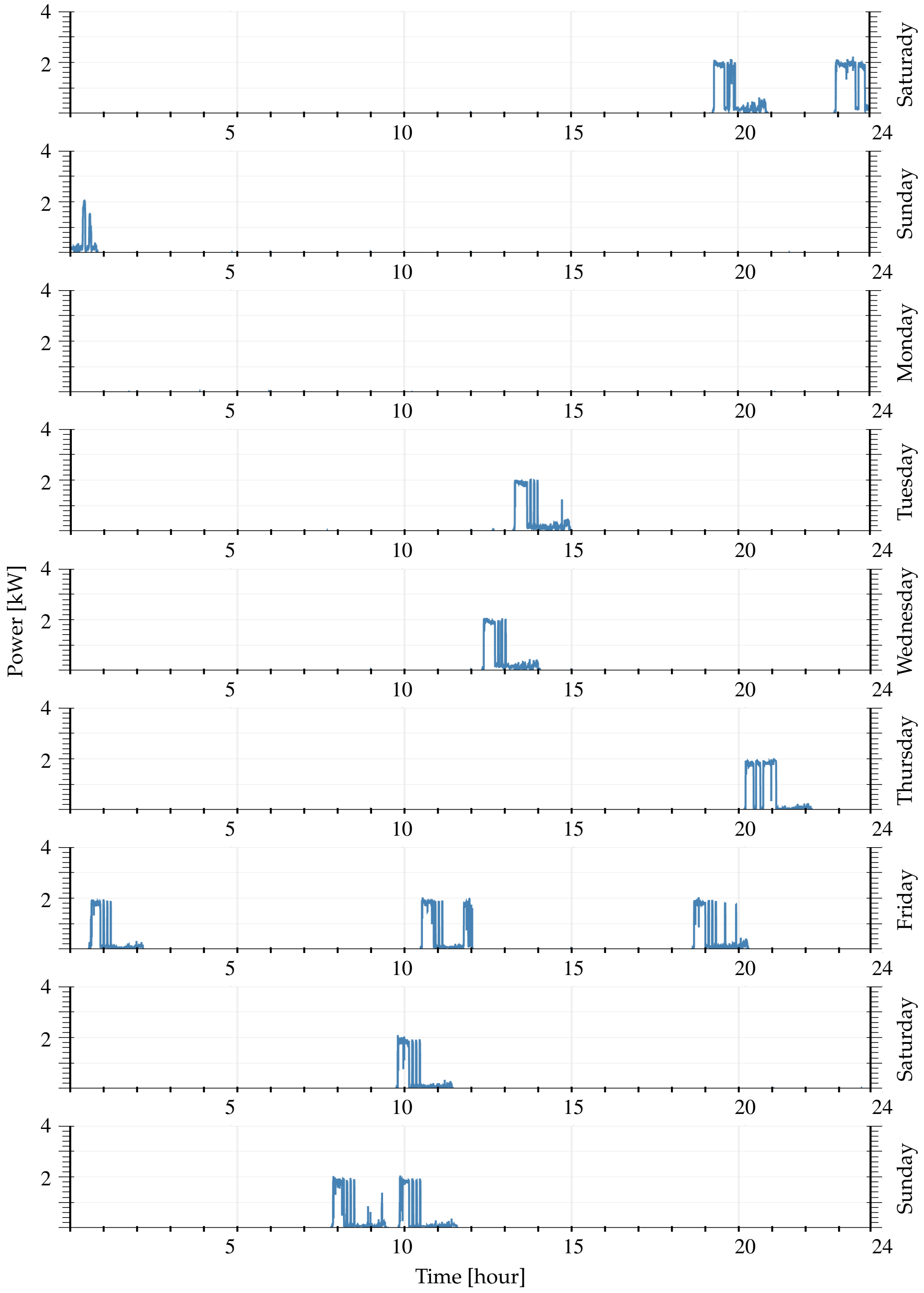


Figure 5.5.: WM disaggregated real power (details in text).

5.7. Conclusion and Outlook

In this chapter, we assessed the feasibility of end-to-end generic deep disaggregation for end-use load monitoring. We focused on the problem of single load extraction for which we proposed a well regularized fully convolutional neural network. The proposed model architecture was evaluated on a variety of load categories from a real-world energy monitoring dataset. The proposed model (with a fixed architecture and set of hyper-parameters) outperforms prior art on the same dataset and showed consistent improvement with respect to the adopted performance measures.

Additionally, we discussed the biasedness of the commonly adopted performance measures in the energy disaggregation research. We then proposed unbiased alternatives for performance assessment that are more flexible and less sensitive to the class imbalance problem.

This work can be extended in the future in various aspects. Exploring deep temporal convolutional architectures (TCN) for sequence-to-sequence modeling is one extension that would additionally investigate the effect of contextual information through their causal convolutional kernels [Bai et al. 2018]. Another, more interesting future work, is the investigation of the loads' cross-dependence and the effect of user intention [Yang and Zhou 2011] from a causal reasoning perspective.

Chapter 6.

Conclusion and Outlook

In this thesis, we presented variants of energy monitoring and disaggregation frameworks ranging from plug-level load monitoring followed by unsupervised energy-disaggregation and finally reaching fully supervised deep-learning-based disaggregation. From all aspects, we proved the feasibility of end-use load monitoring and disaggregation and proposed reliable frameworks and models that were experimentally validated on real-word residential and commercial datasets.

The varying viewpoints of energy disaggregation (from an unsupervised framework to a semi-supervised one and finally reaching a fully supervised model) showed that the energy disaggregation task can be adaptively addressed based on the availability of domain knowledge, training data, or both.

Additionally, a plug-level monitoring framework showed a superior end-load identification level. It further showed a high level of robustness to sampling rate variations and stable performance over time.

In addition to all possible future works detailed at the end of each chapter, we further propose the following.

As it was shown in Chapter 2, a fully supervised model, on its own, is expected to very sensitive to the size and availability of externally labeled data. Therefore, and while this thesis presented each disaggregation model as a standalone component, a practical energy disaggregation system comprising all proposed components is expected to be notably adaptive to the availability of training data. Such a system would mostly rely on the unsupervised (even though conservative) component whenever no training data is provided, leverage a semi-supervised model upon scarce available of labeled examples, and finally reach the fully supervised scenario upon adequate acquisition of training data.

A second, and very promising, future work is accounting for uncertainties amongst all components where not only the system designers can express their uncertain beliefs on model parameters, but also each disaggregation component can express and propagate uncertainties in their predictions.

Appendix A.

Event Detection Results

Figures A.2 to A.26 depict examples of the event detection and feature extraction stages of our unsupervised disaggregation framework applied to sub-metered channels of the commercial dataset. The figures show the real-power signal (red curve) for one week of measurements, the event detections (highlighted in blue), and the extracted features (in the lower right `dP-dQ` plane).

Input signals for event detection (and feature extraction) are the real and reactive power signals sampled at 1 Hz frequency. Note that, some channels (that is, Figures A.2, A.13, A.14, A.19, A.23 and A.26) comprised loads with non-detectable state changes for varying reasons. Figure A.26, for instance, depicts the signal of an always-on load with a slight (around 3 Watts) increase in the power draw during a working hours of a day. Figures A.2, A.13 and A.19 show abrupt changes in the aggregate power signals that are far below our minimal threshold (that is, 10 Watt for these experiments). Finally, loads under channels of Figures A.3 and A.23 were not operational throughout the monitoring period.

Worth mentioning as well is that, some channels (e.g. Figures A.17 and A.21) were monitoring indeed a single end-use load, while others (e.g. Figure A.20) monitored a sub-set of loads.

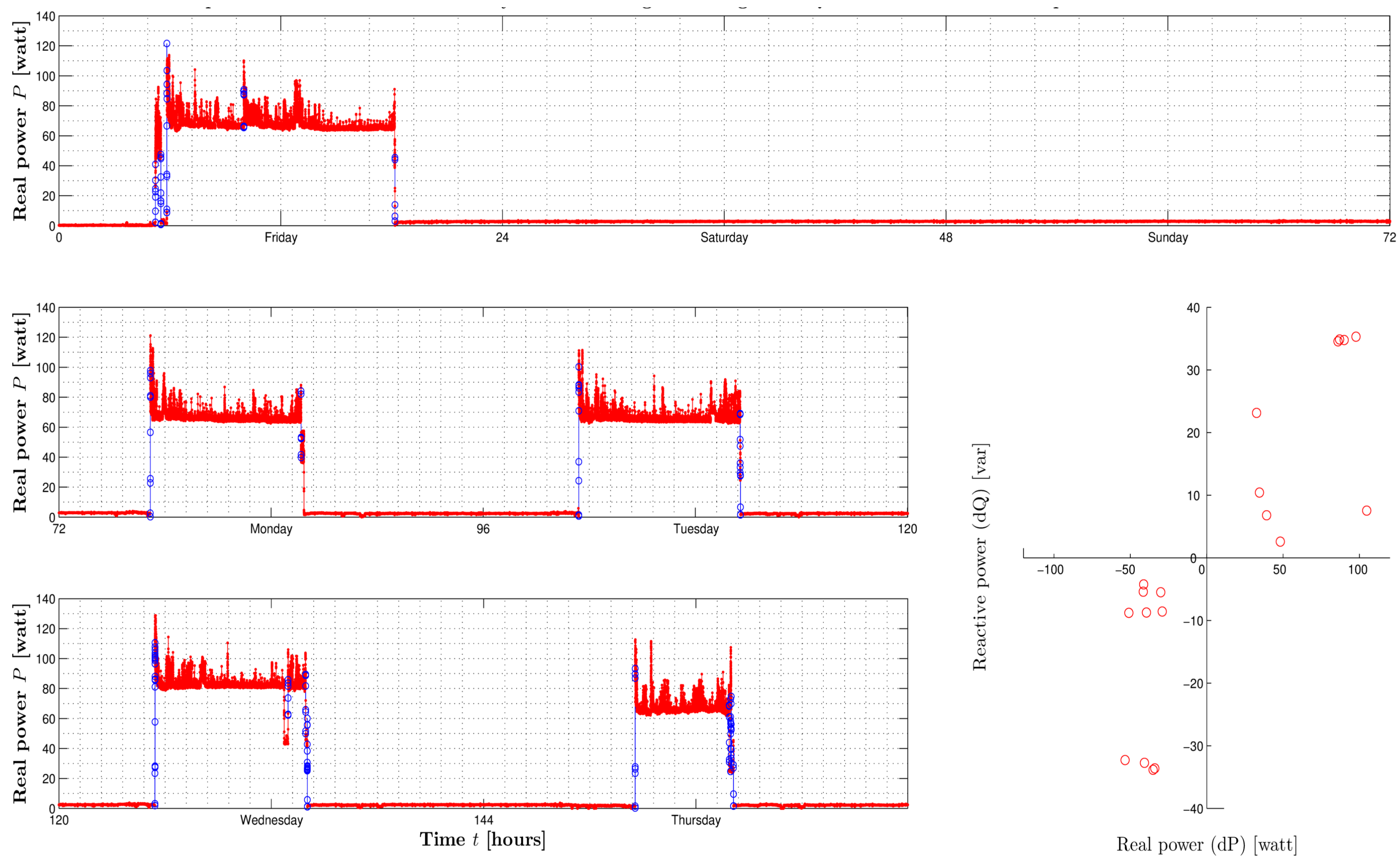


Figure A.1.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

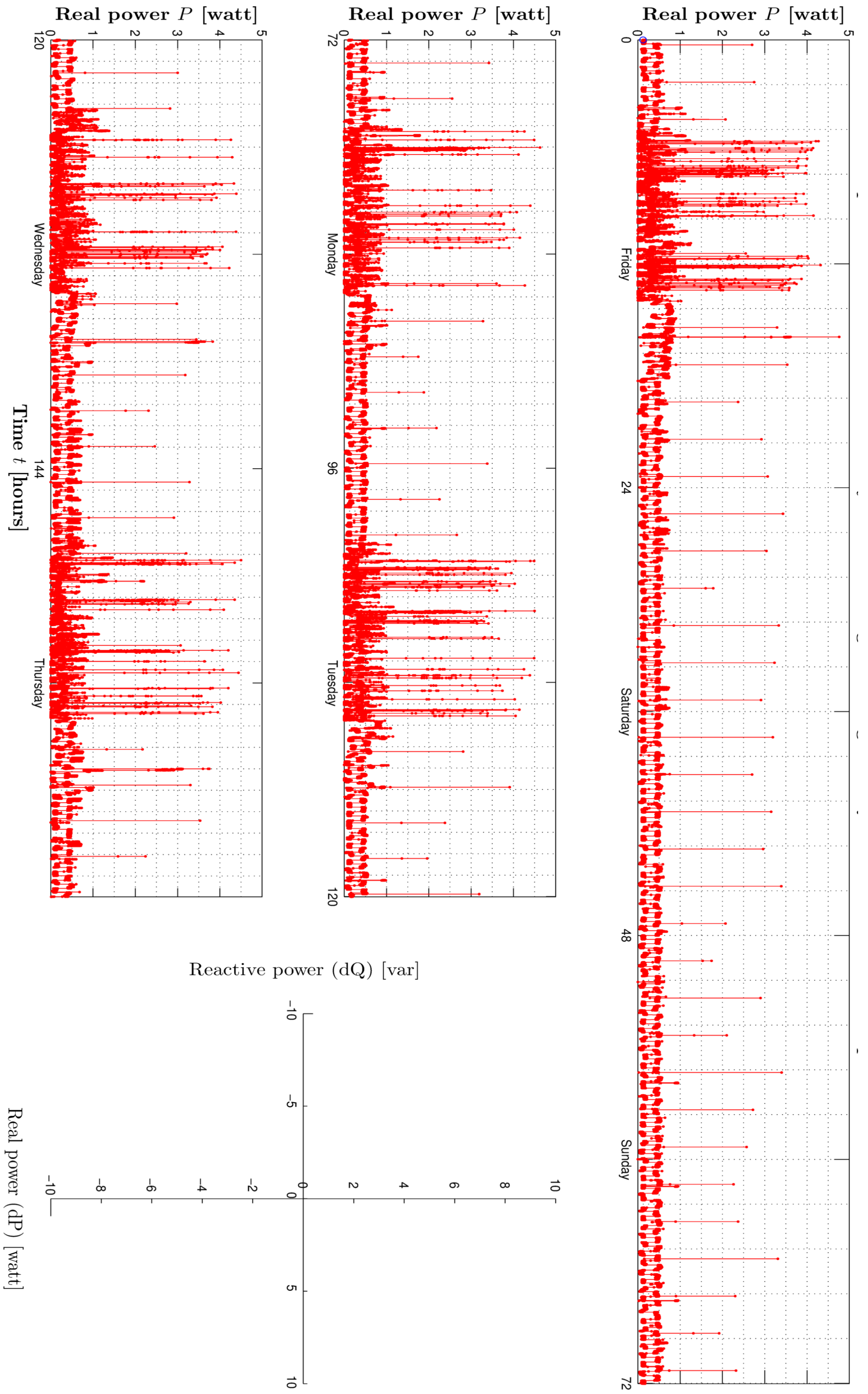


Figure A.2.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

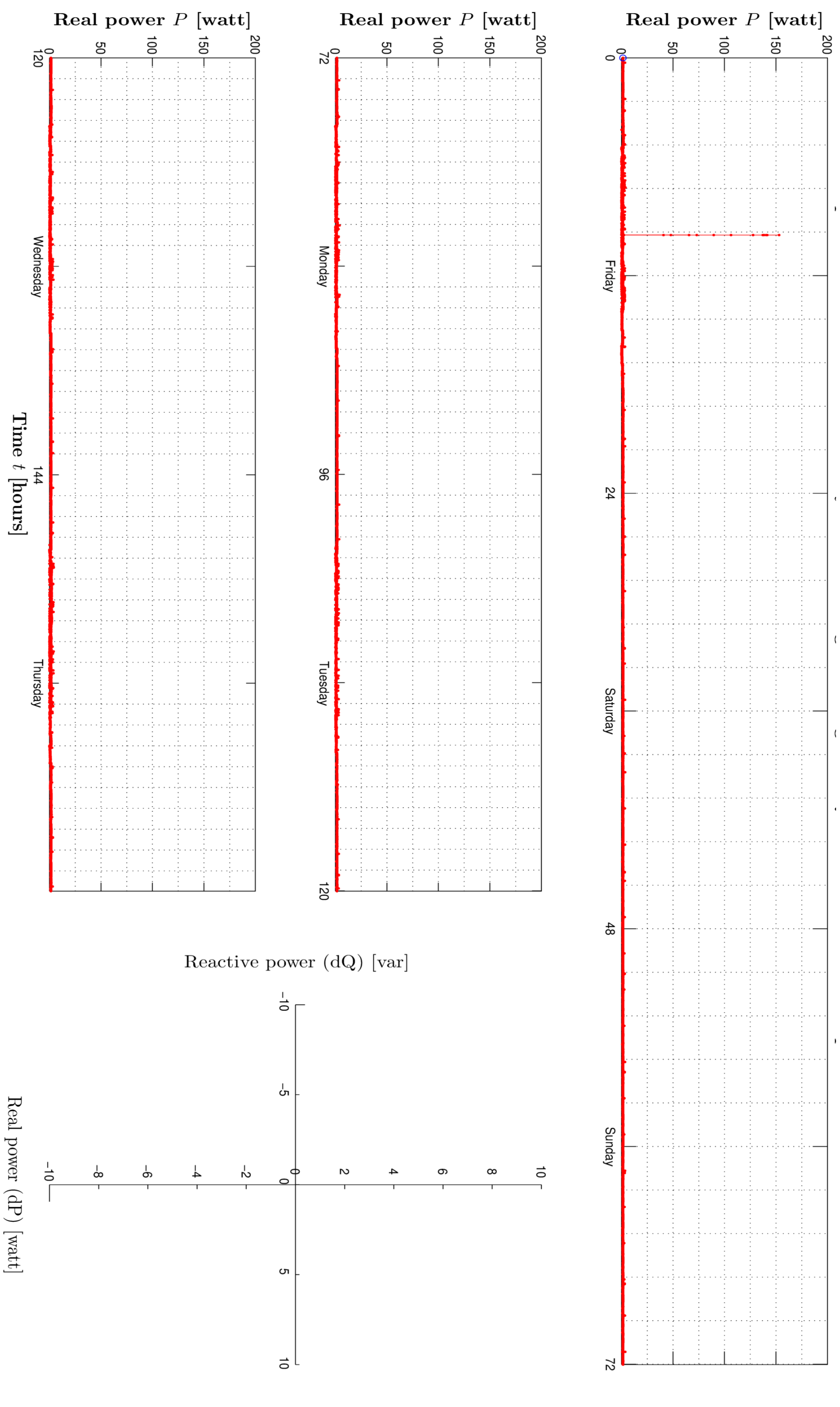


Figure A.3.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

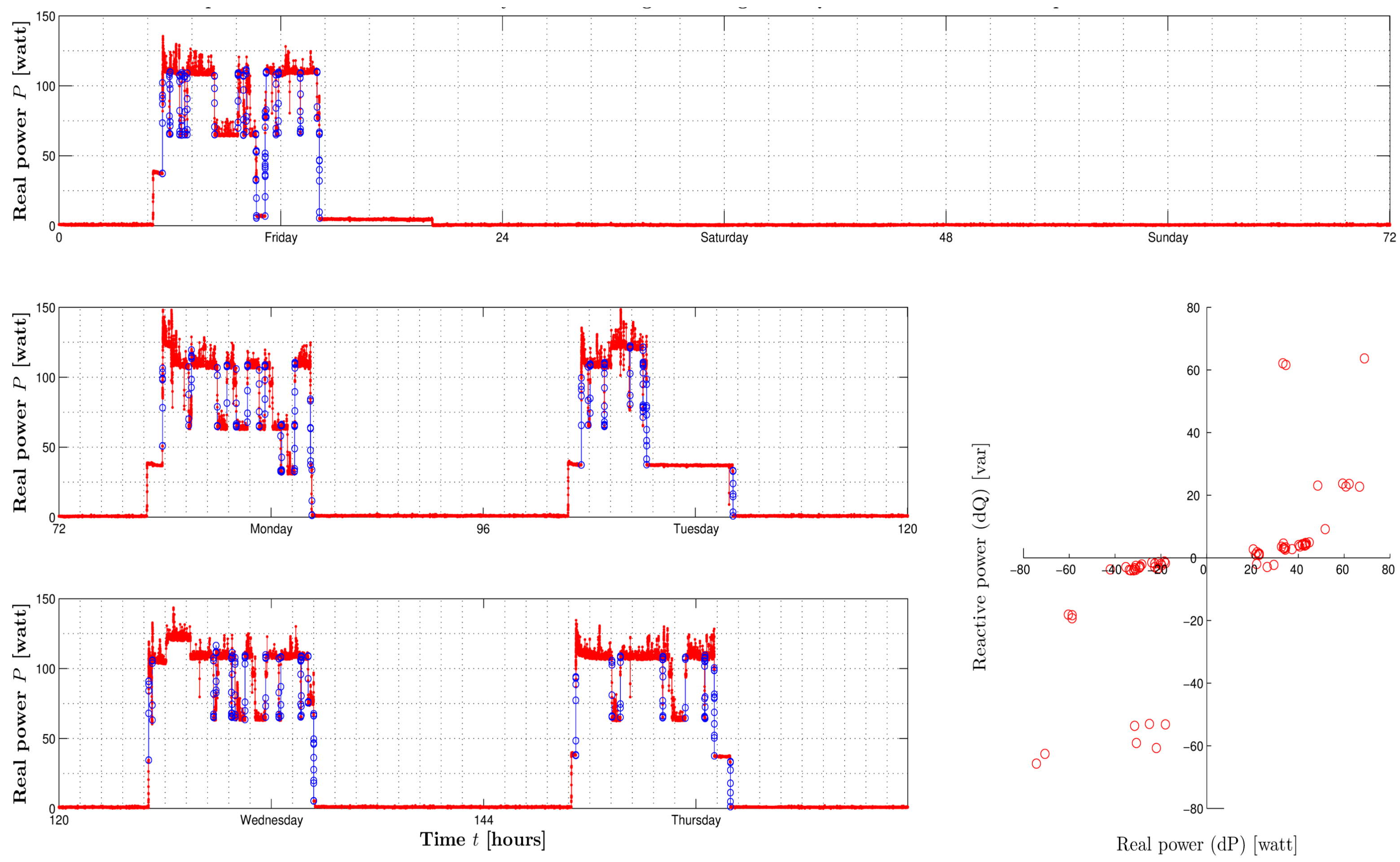


Figure A.4.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

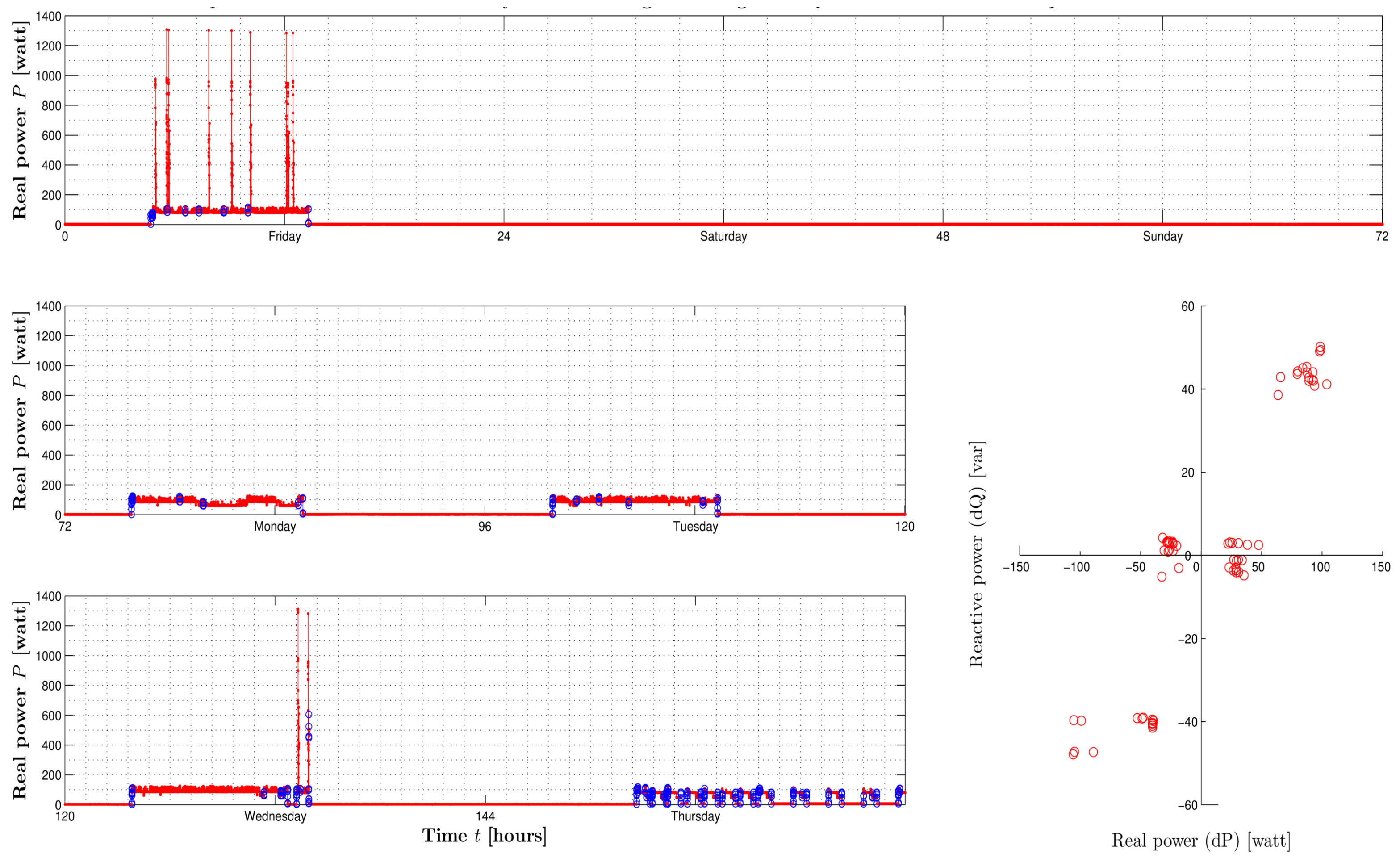


Figure A.5.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

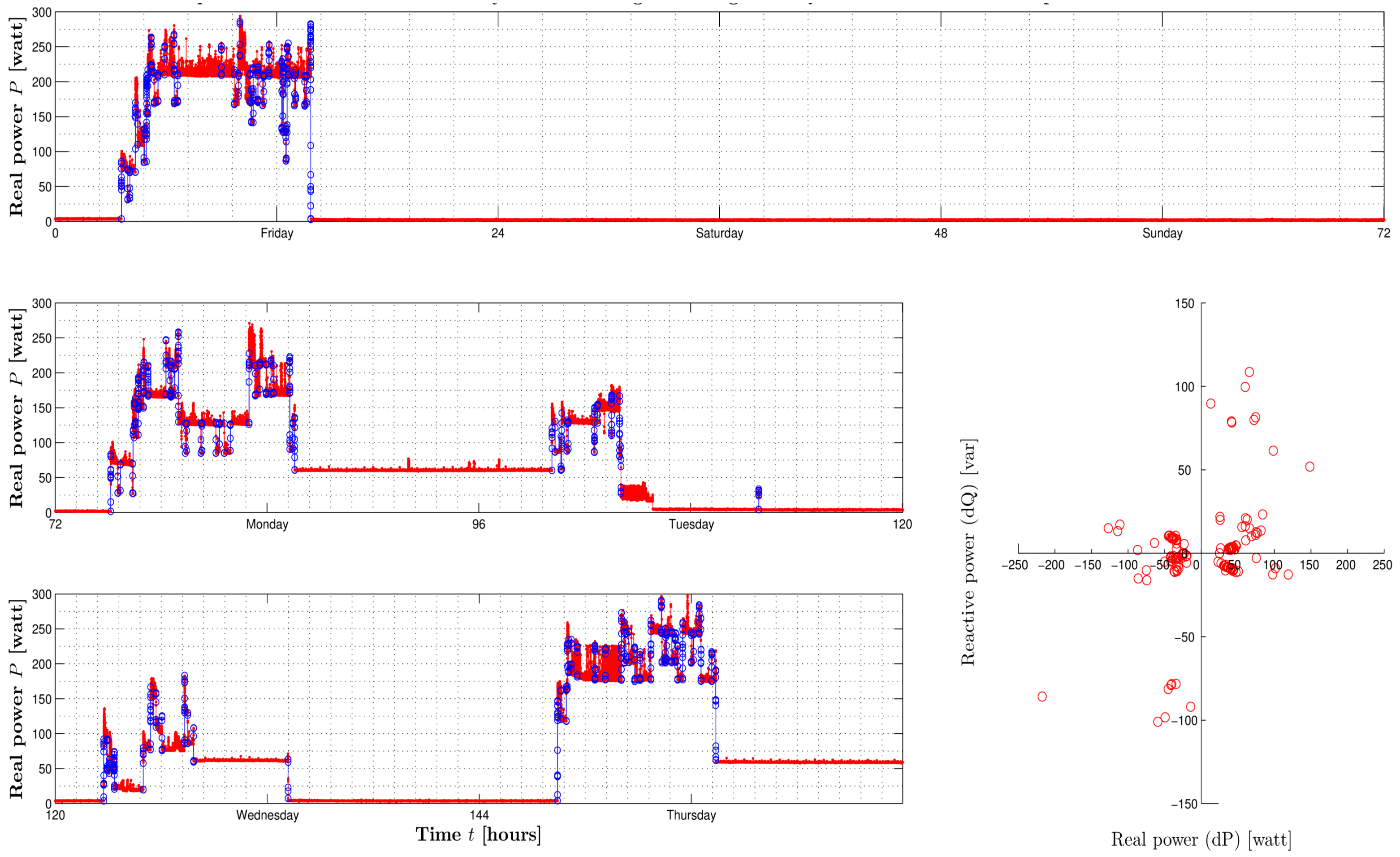


Figure A.6.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

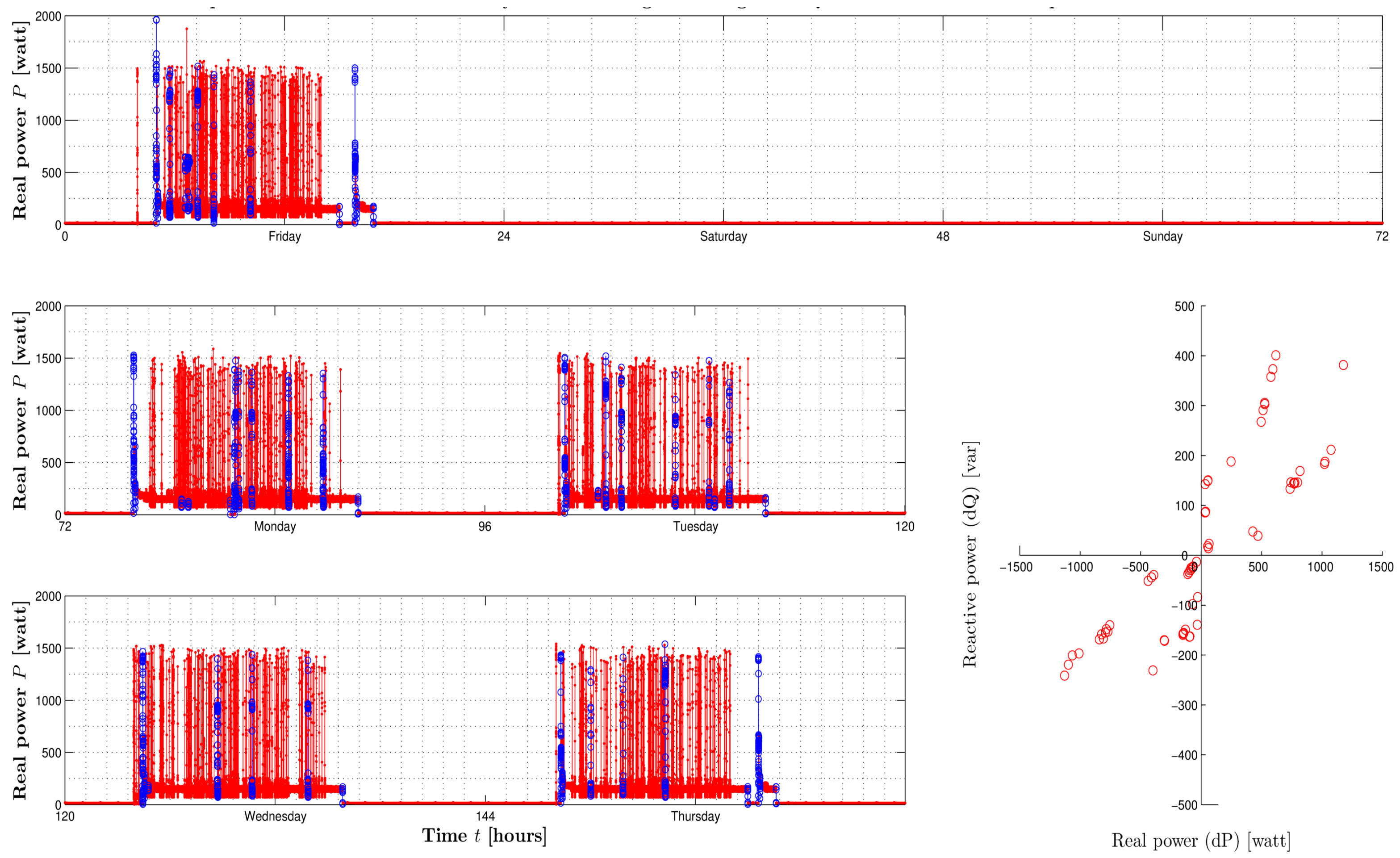


Figure A.7.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

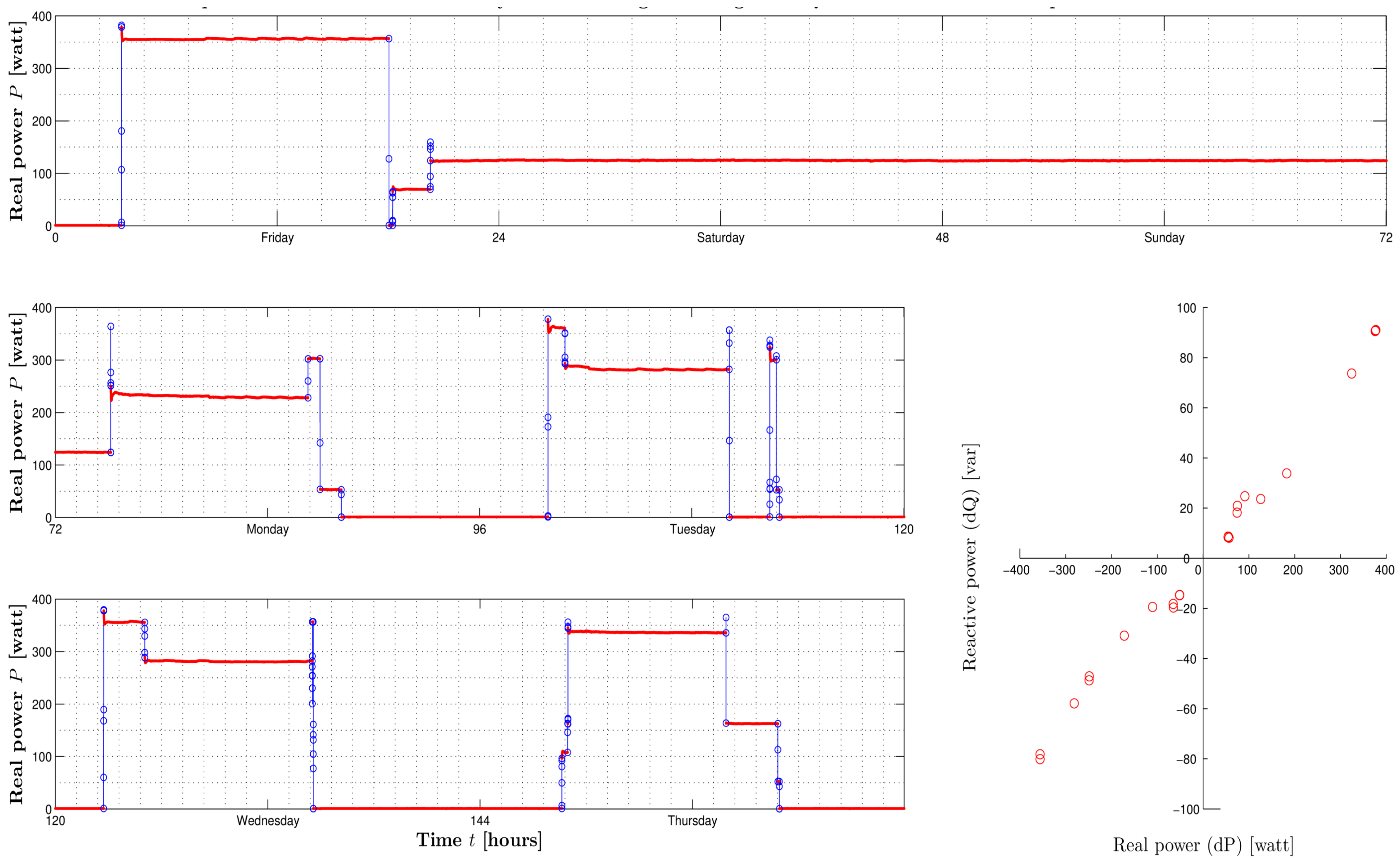


Figure A.8.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

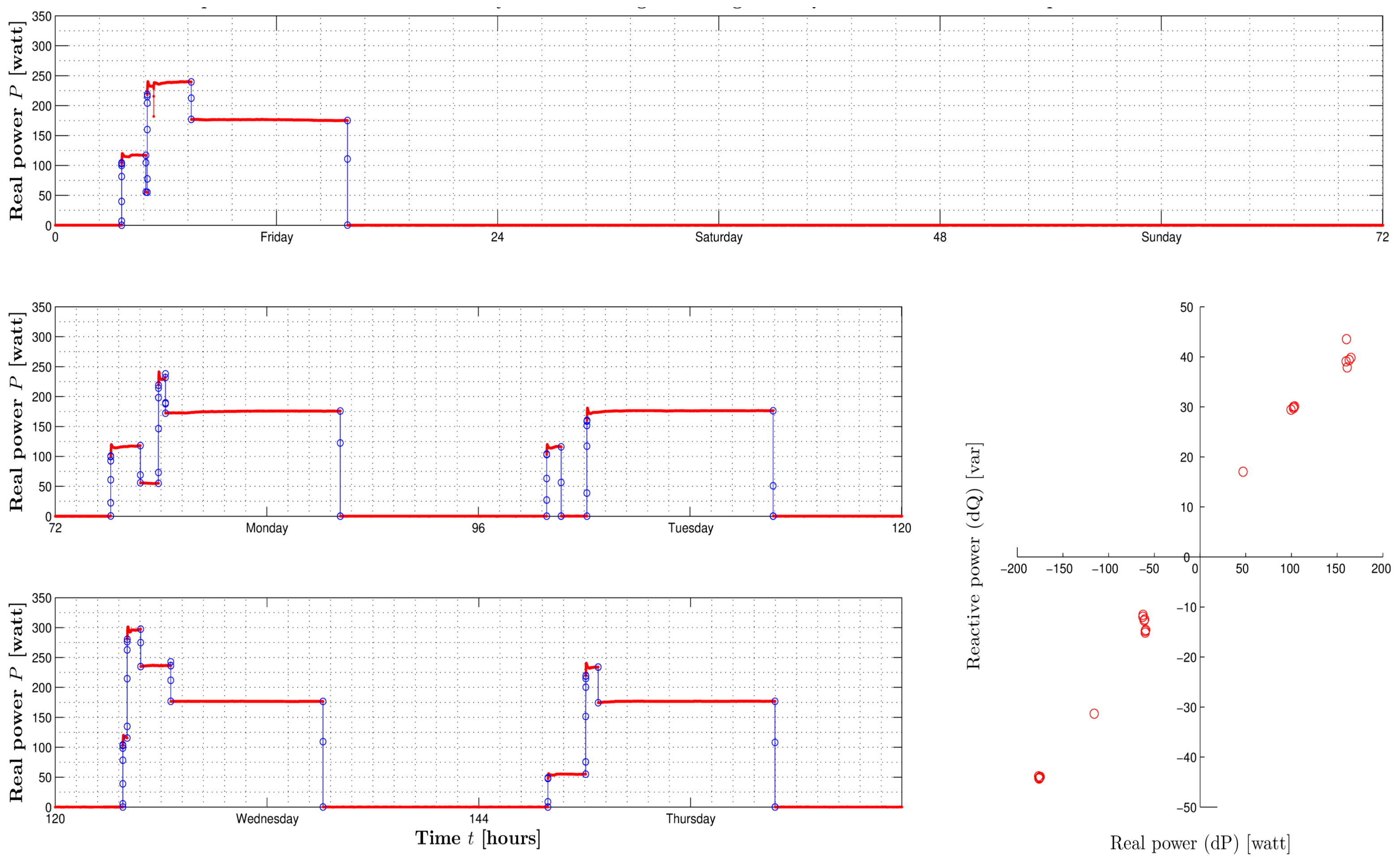


Figure A.9.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

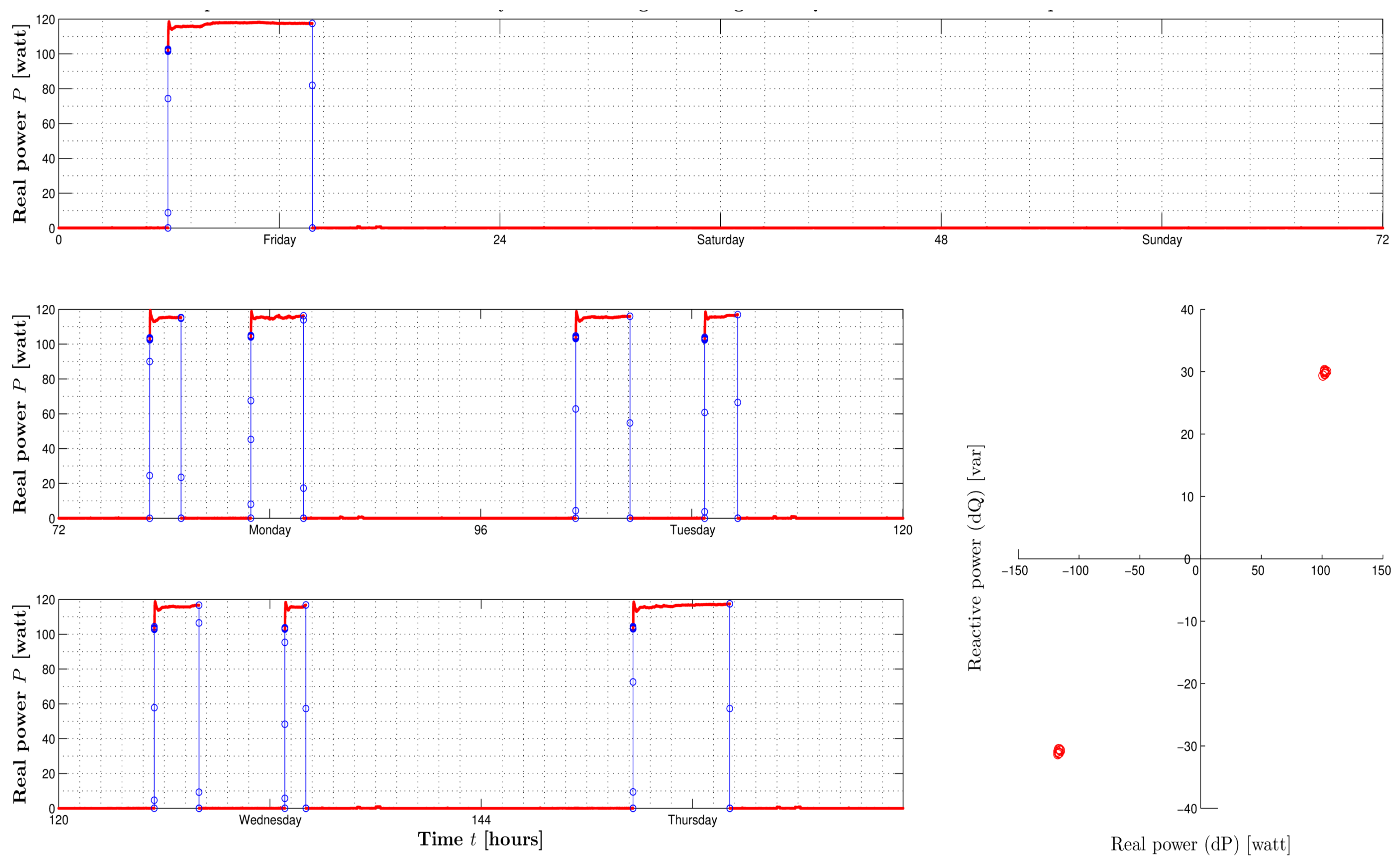


Figure A.10.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

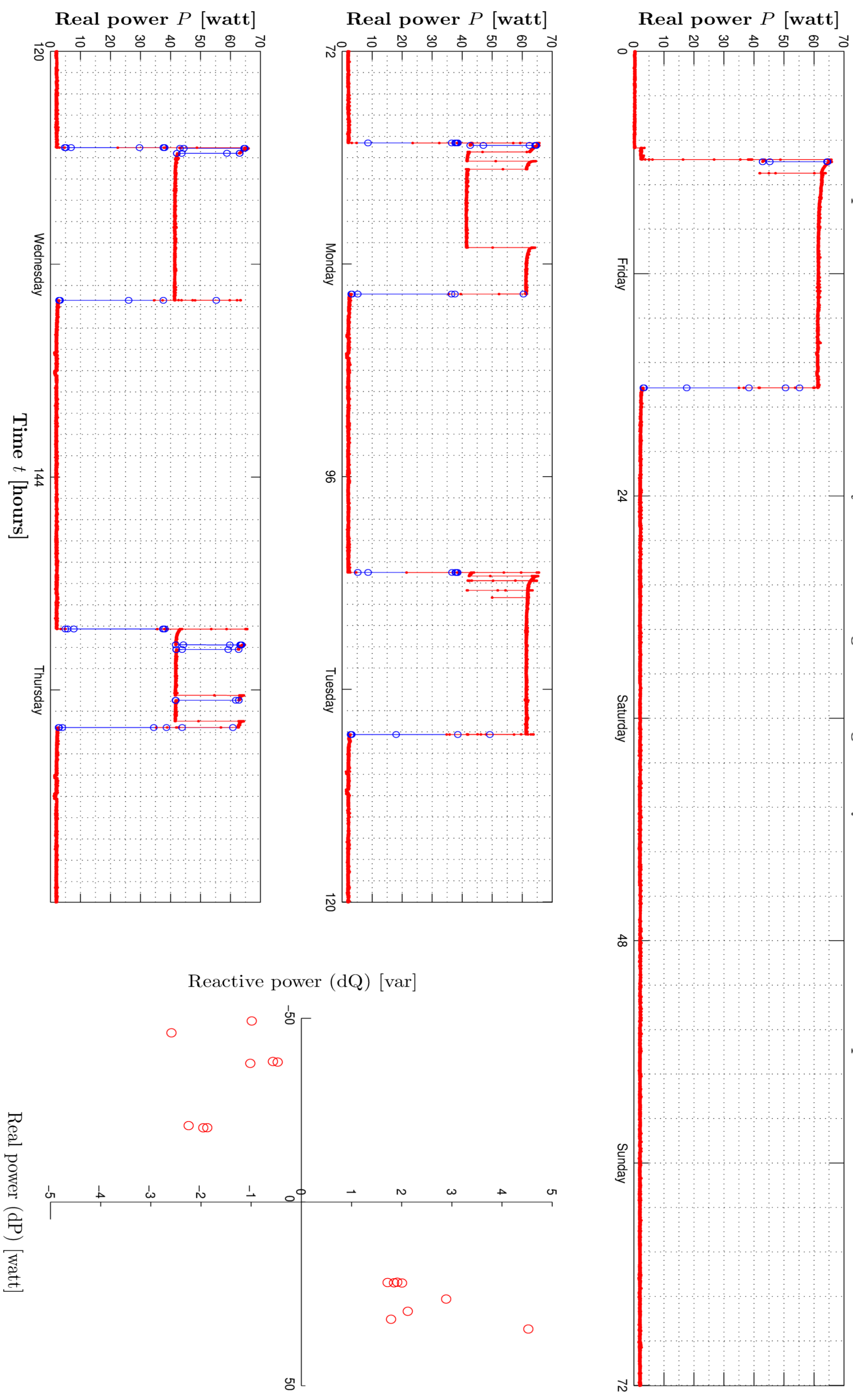


Figure A.11.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

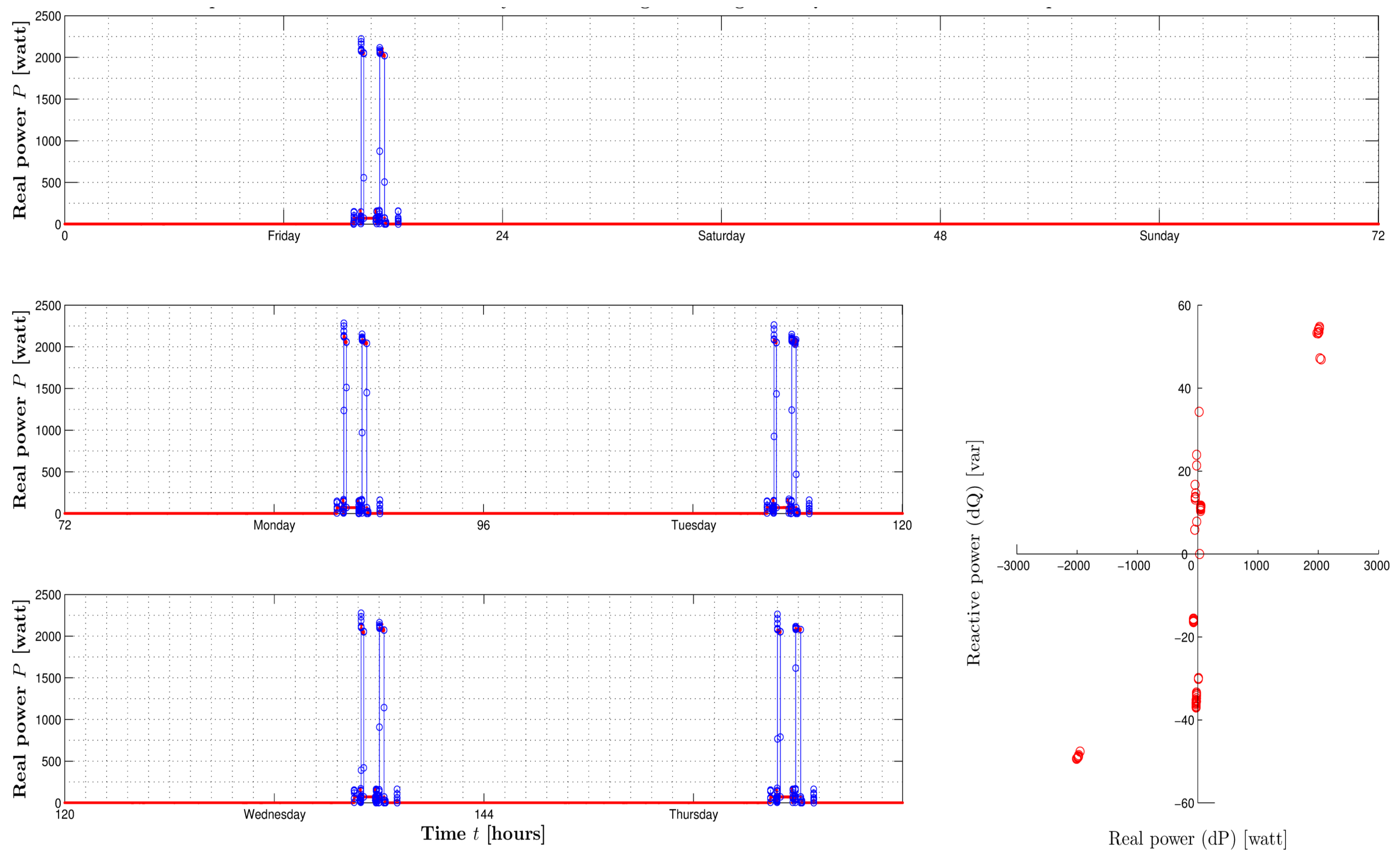


Figure A.12.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

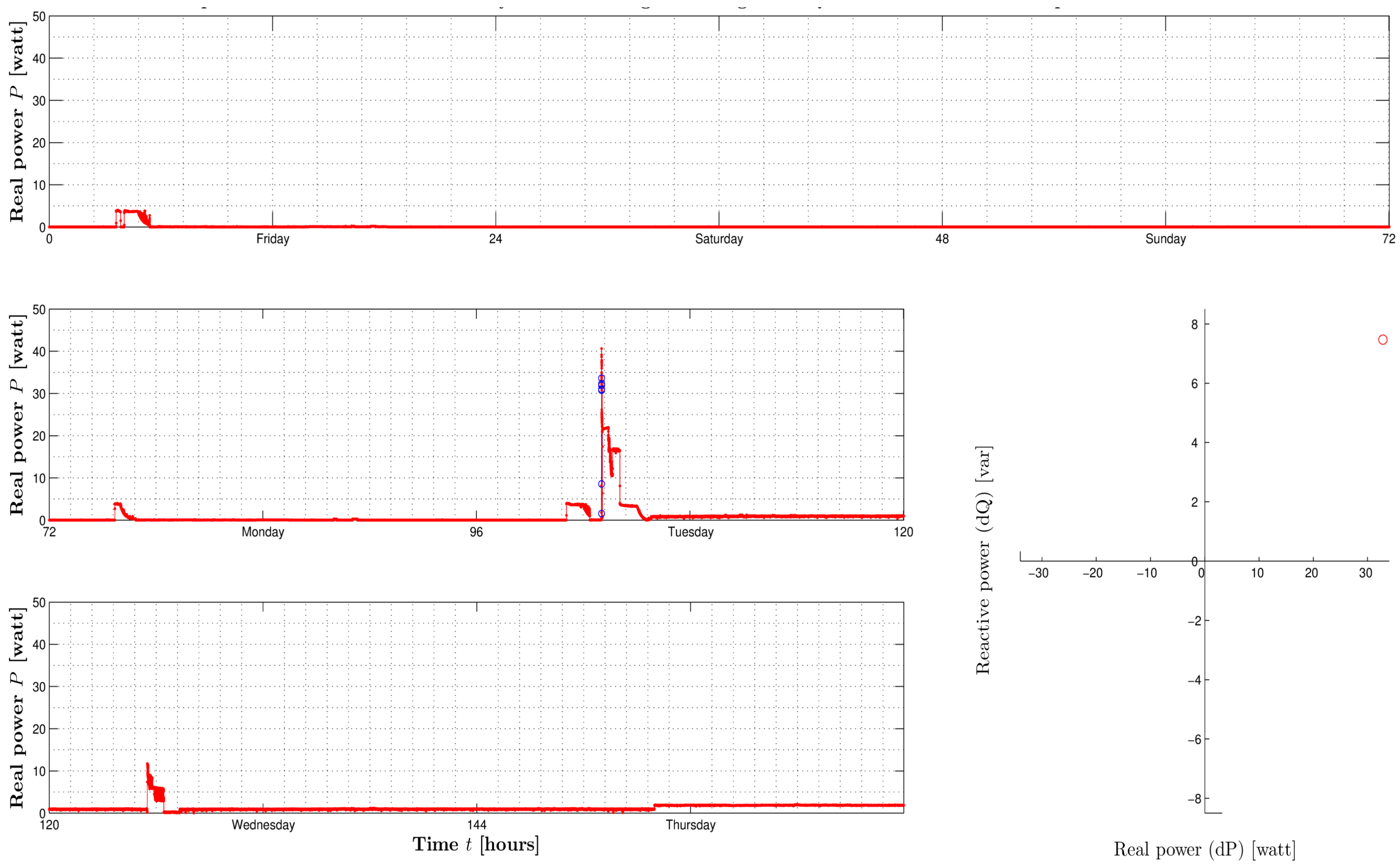


Figure A.13.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

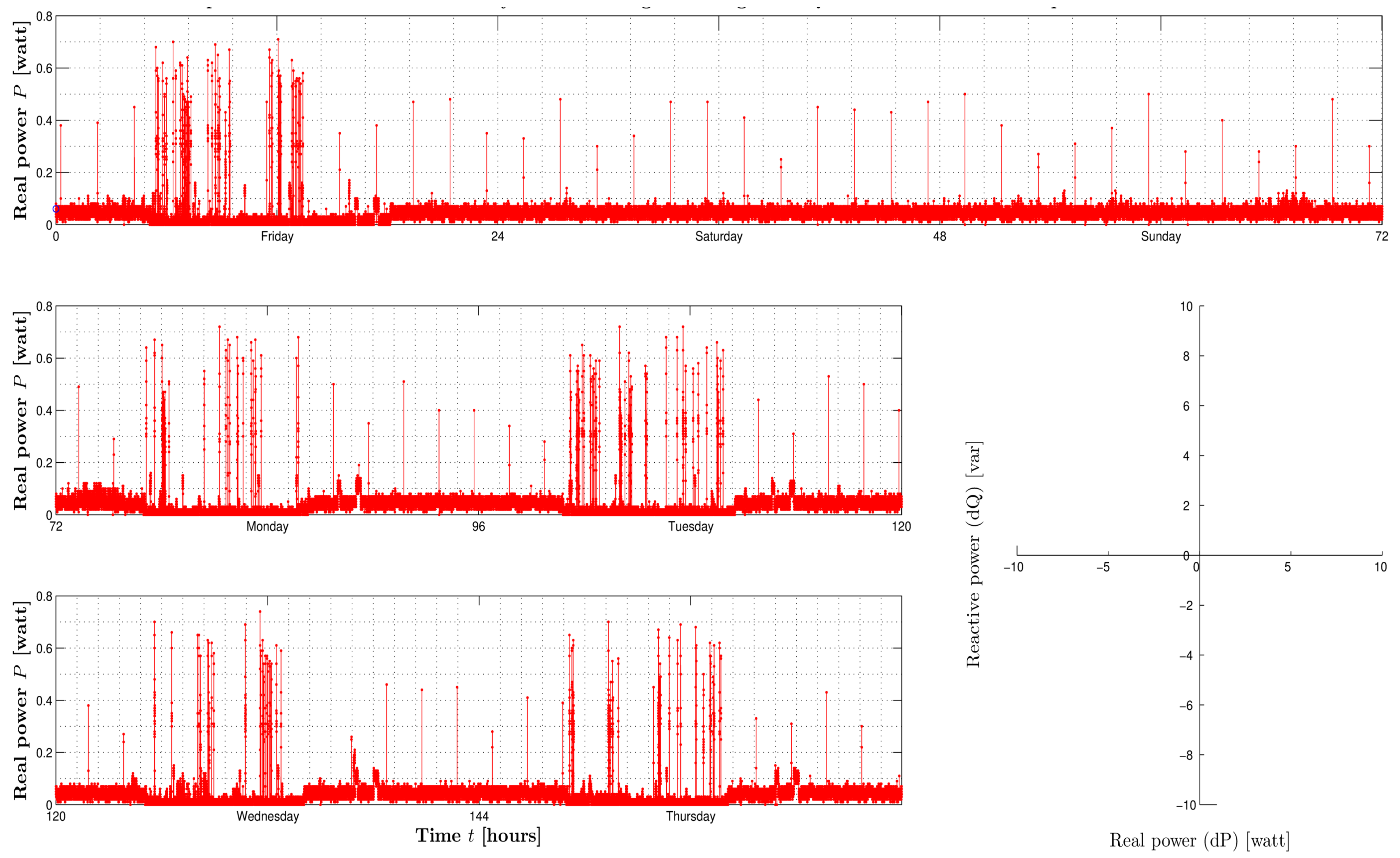


Figure A.14.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

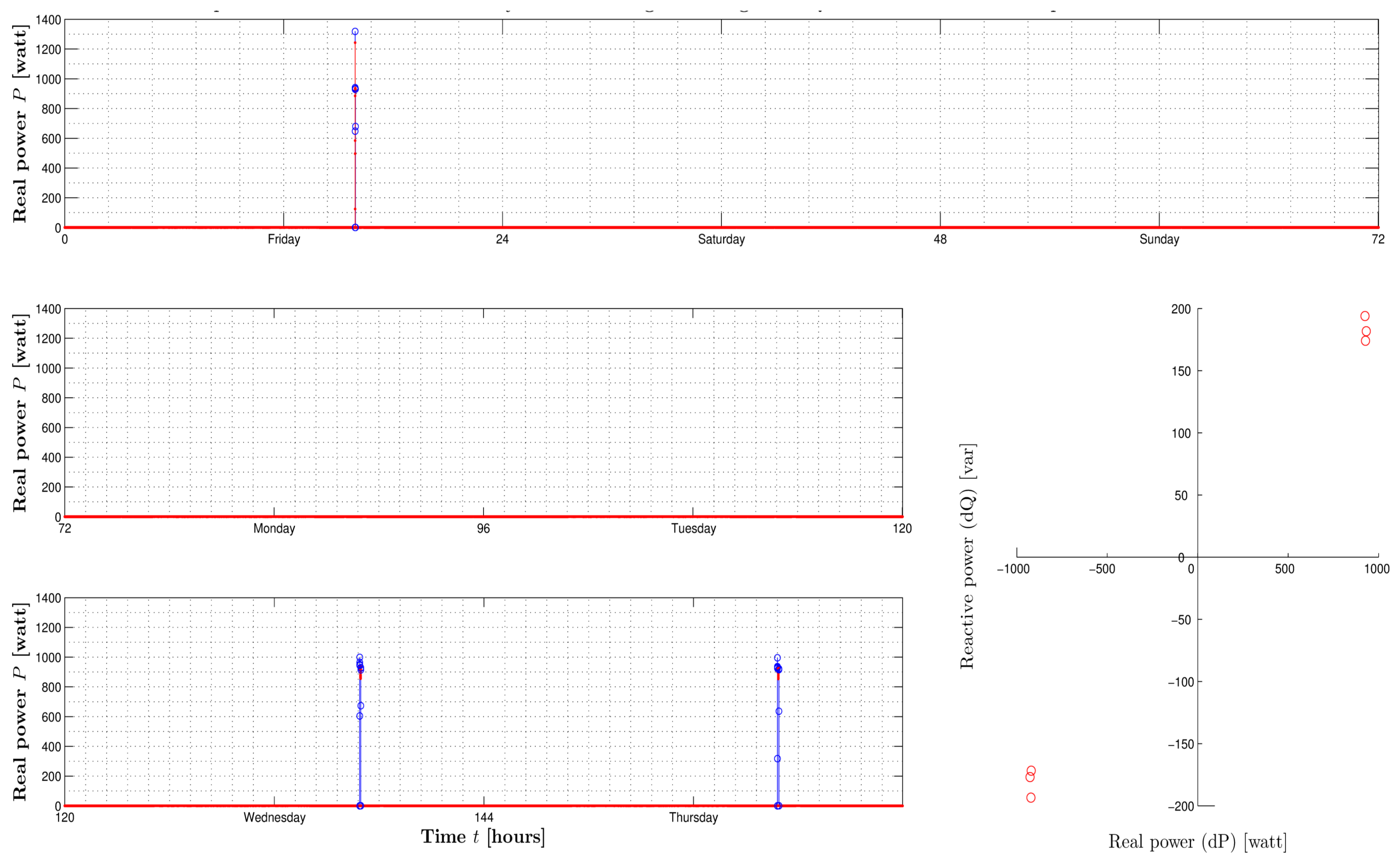


Figure A.15.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

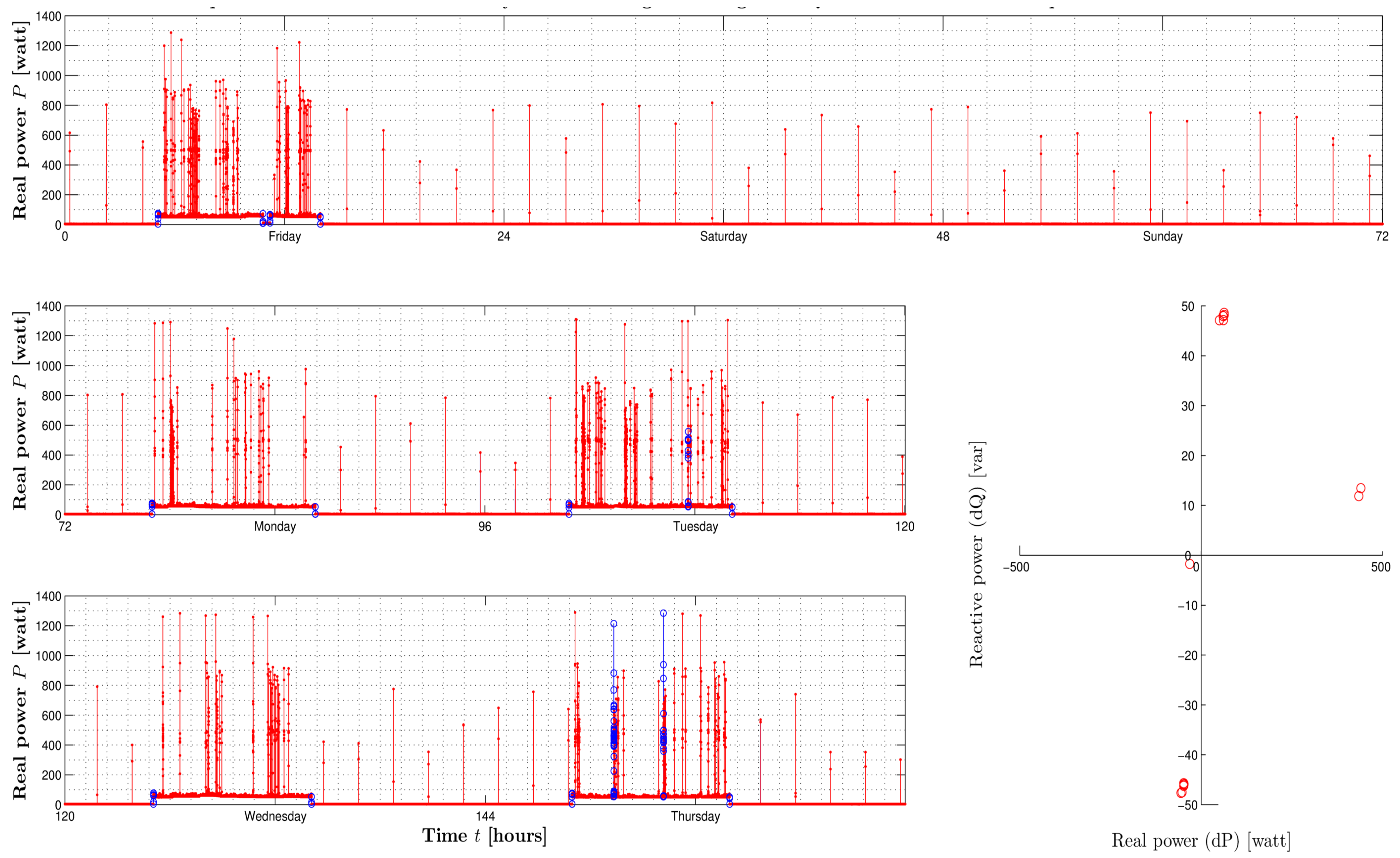


Figure A.16.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

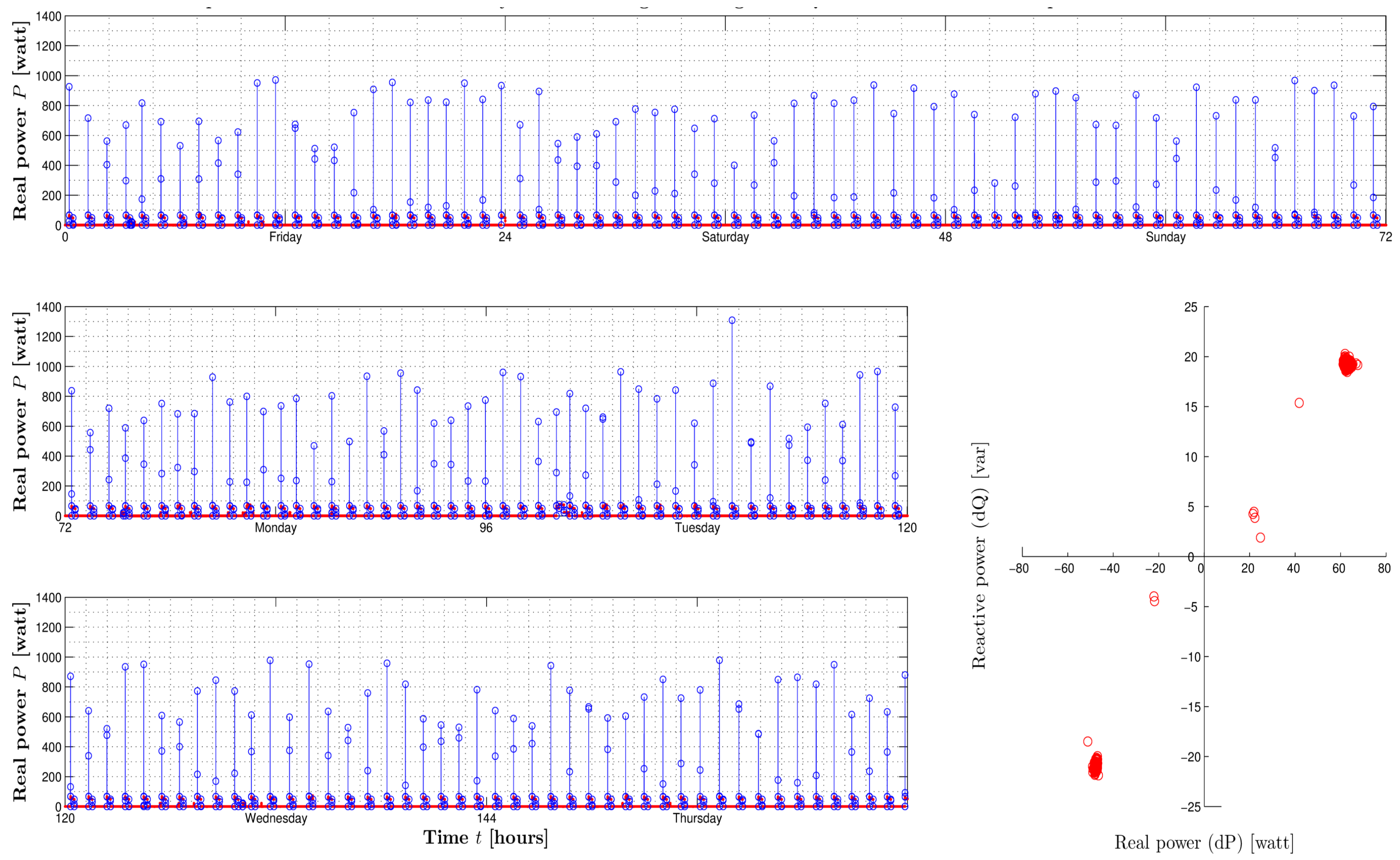


Figure A.17.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

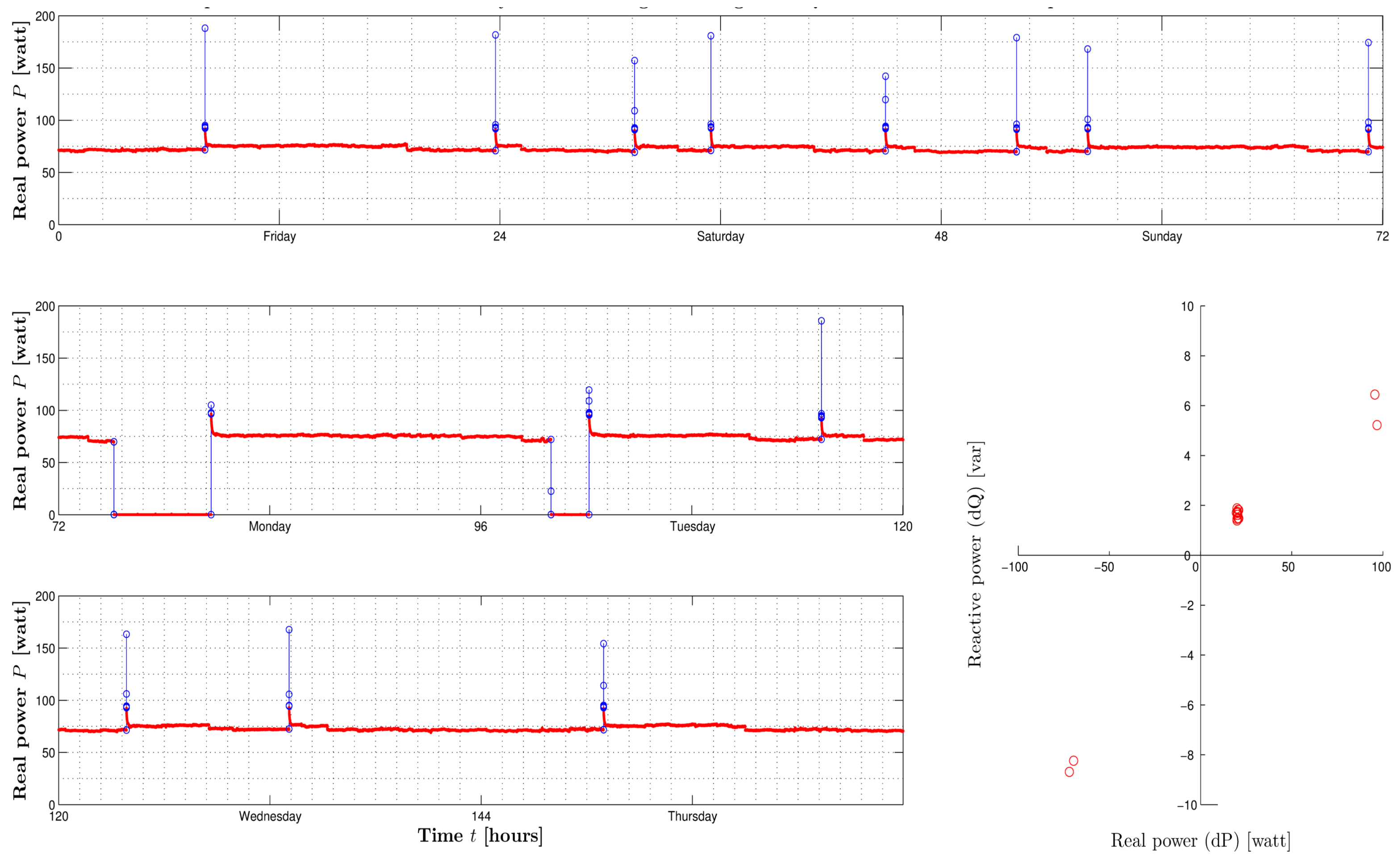


Figure A.18.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

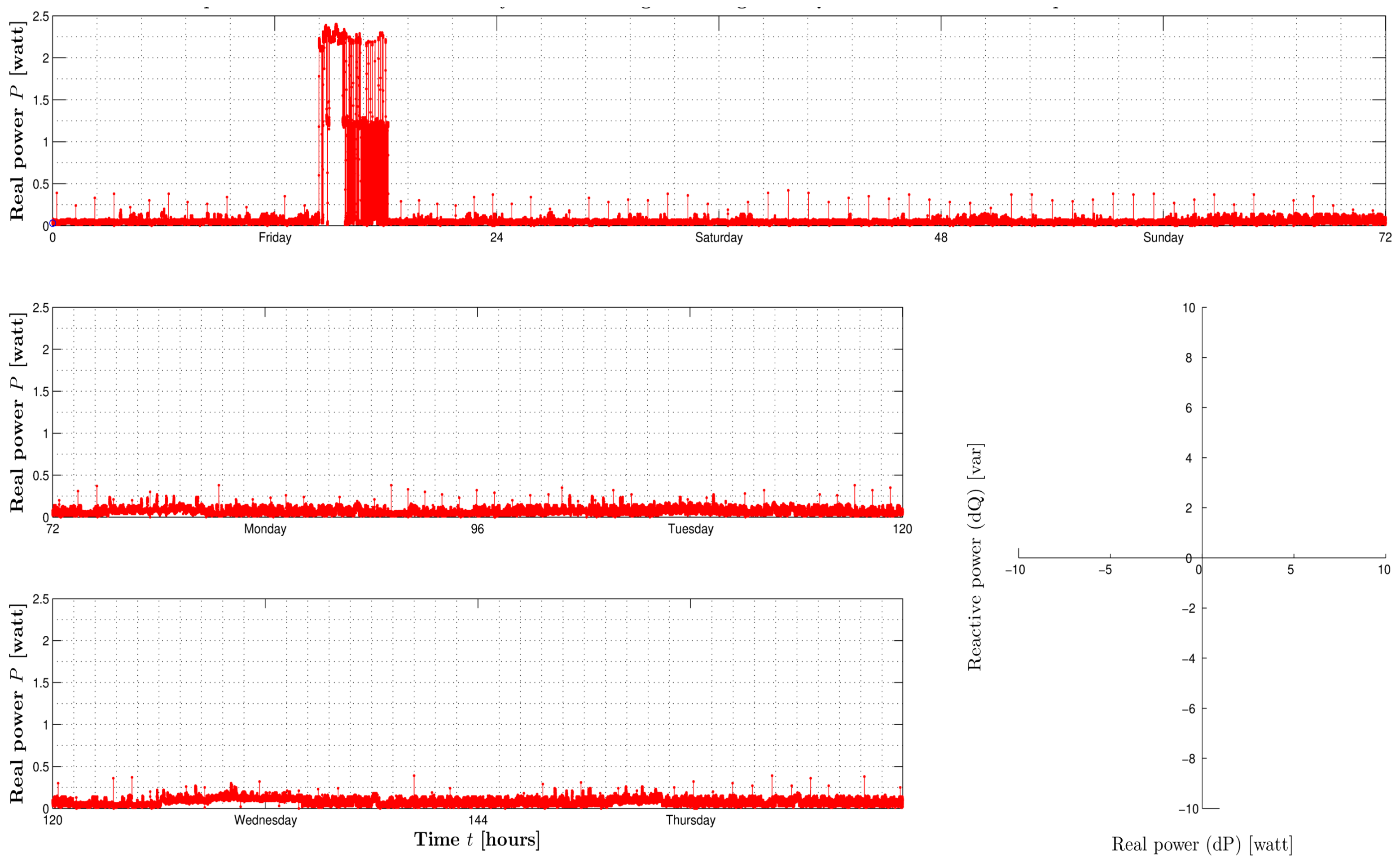


Figure A.19.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

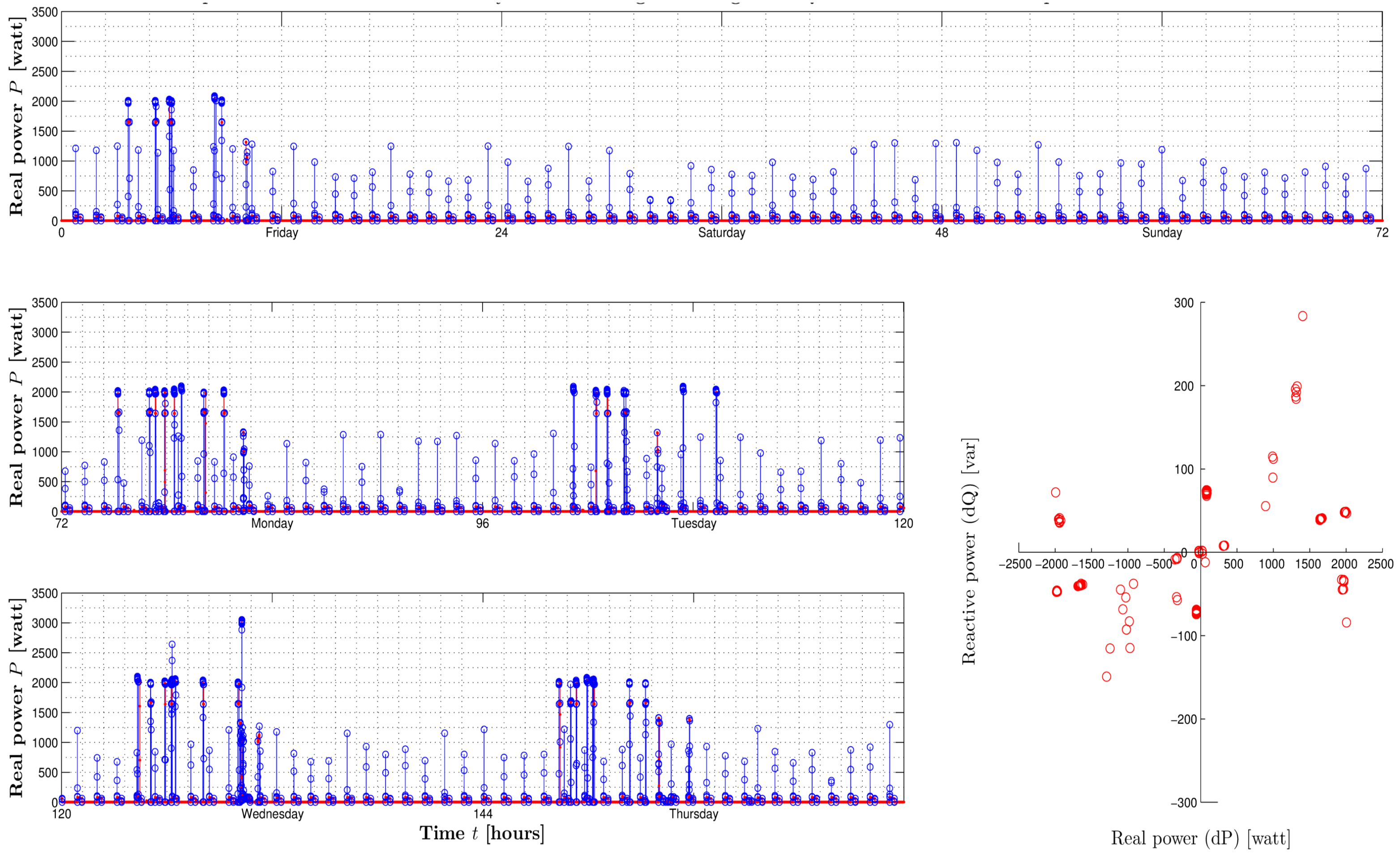


Figure A.20.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

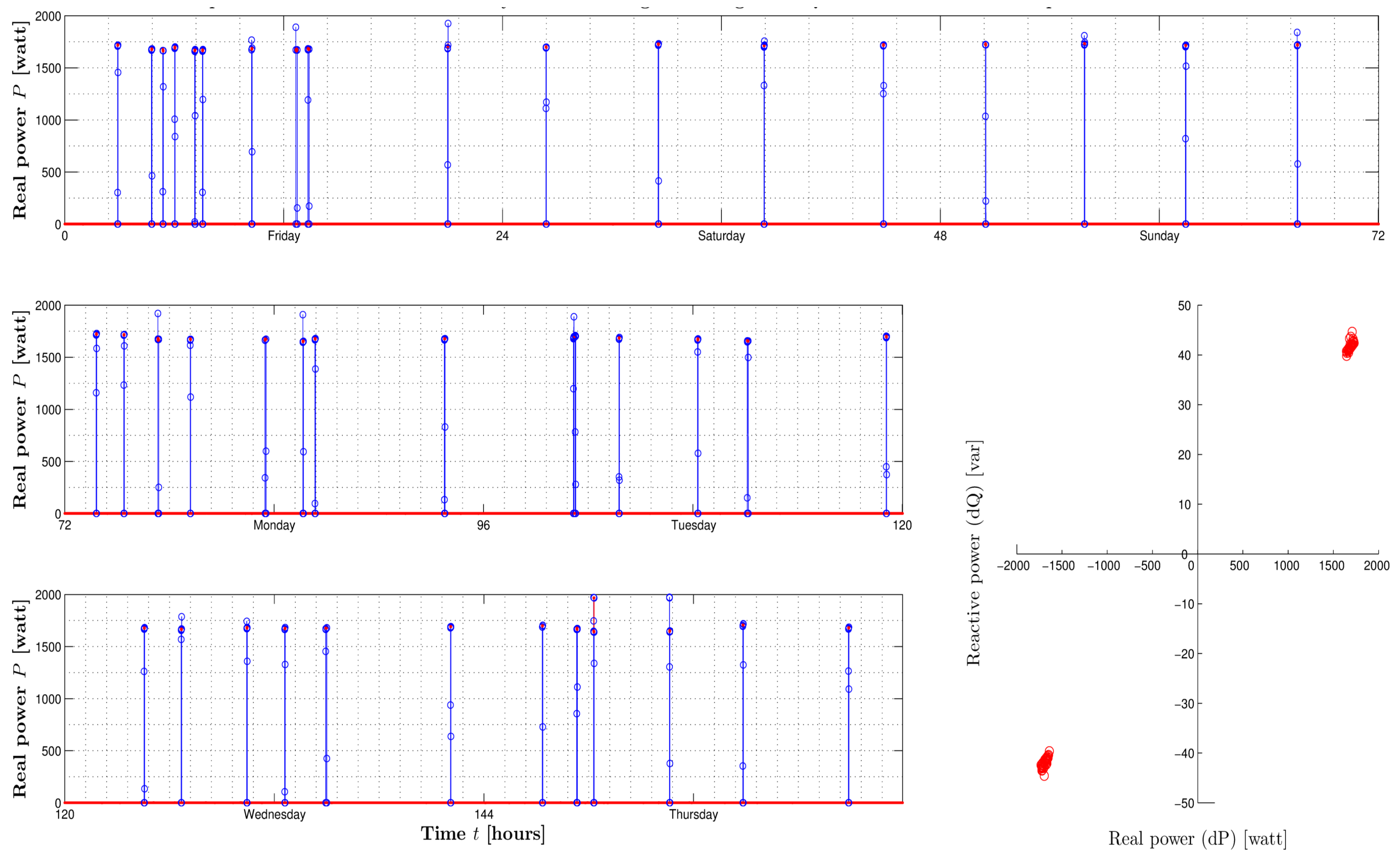


Figure A.21.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

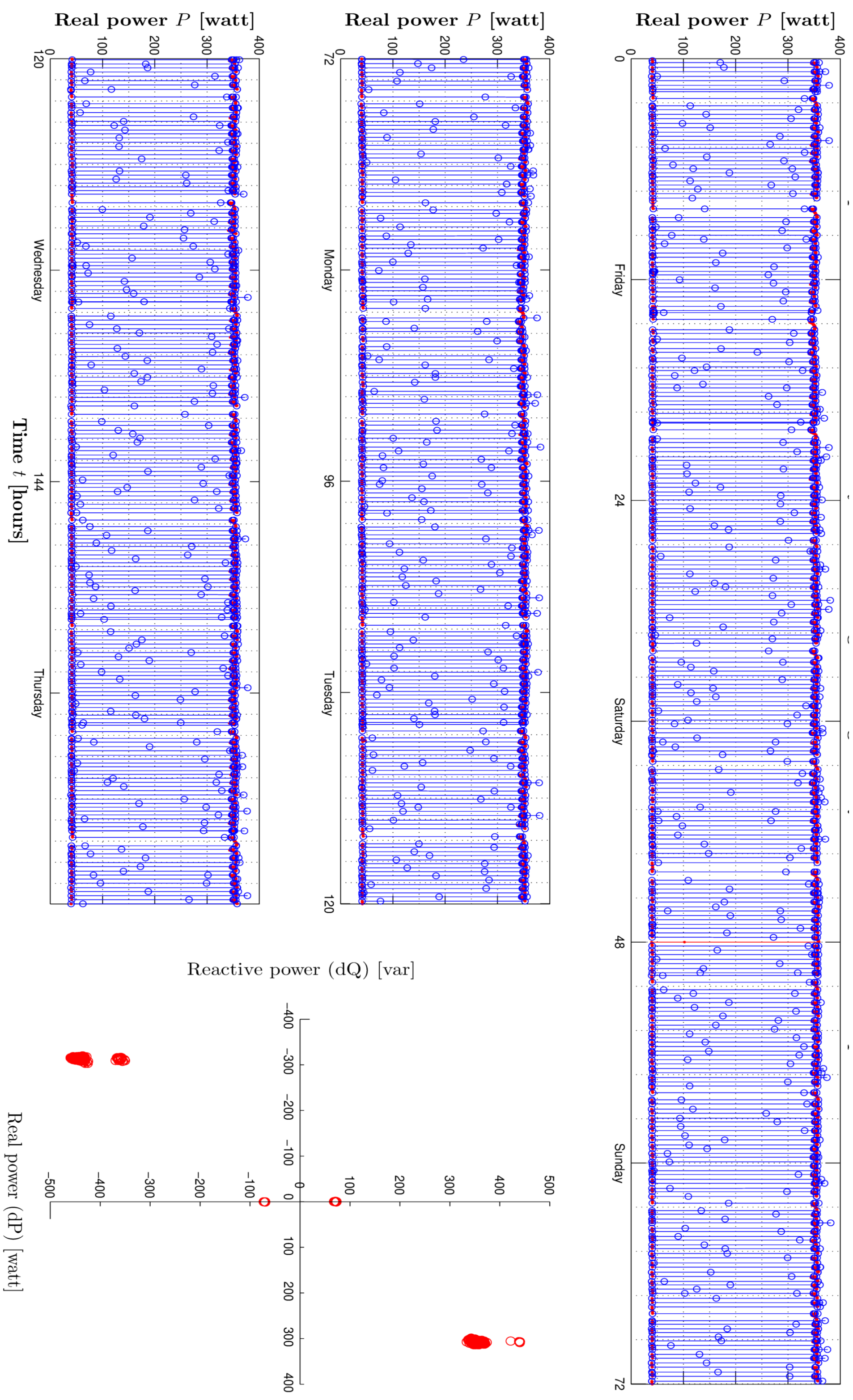


Figure A.22.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

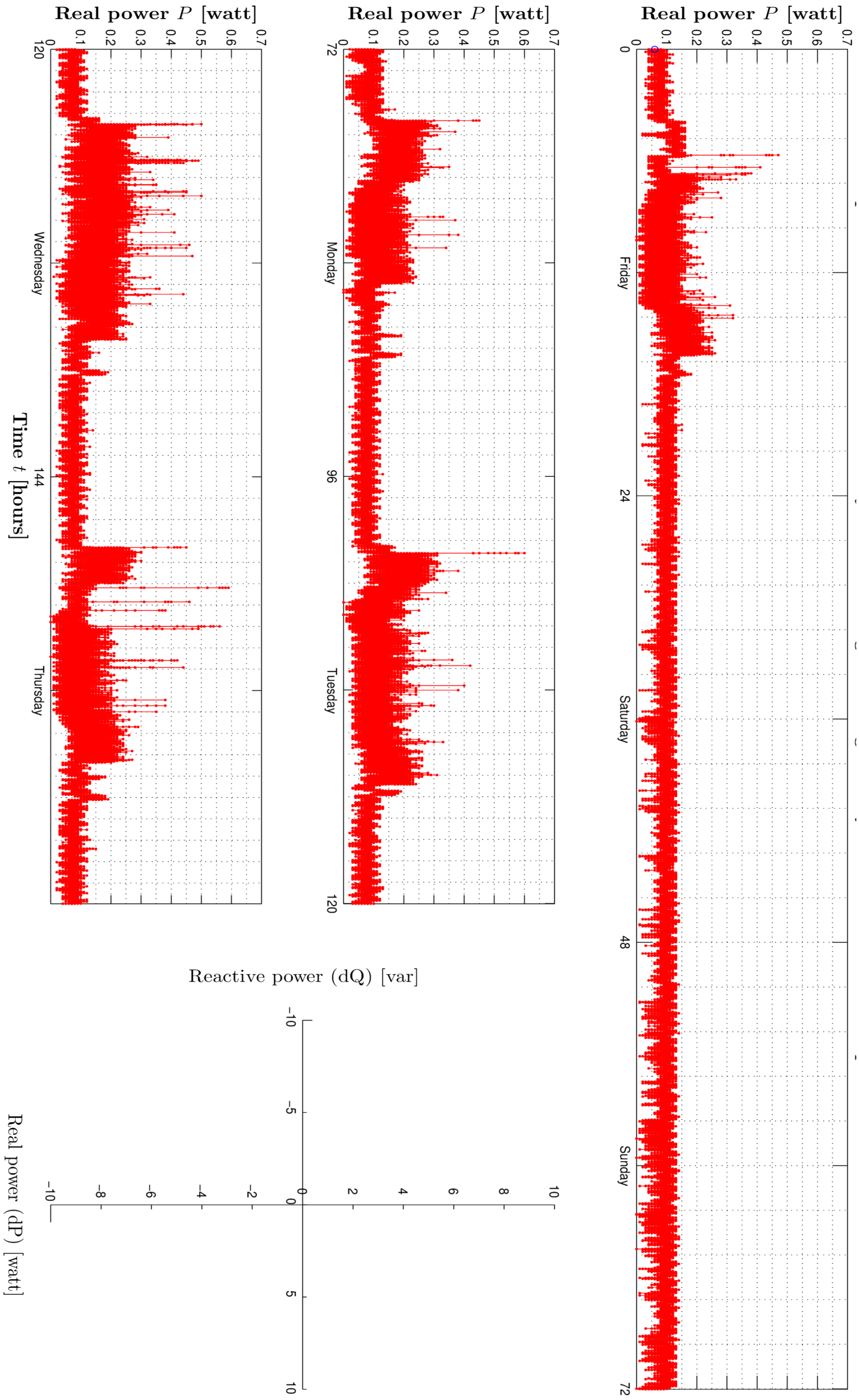


Figure A.23.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

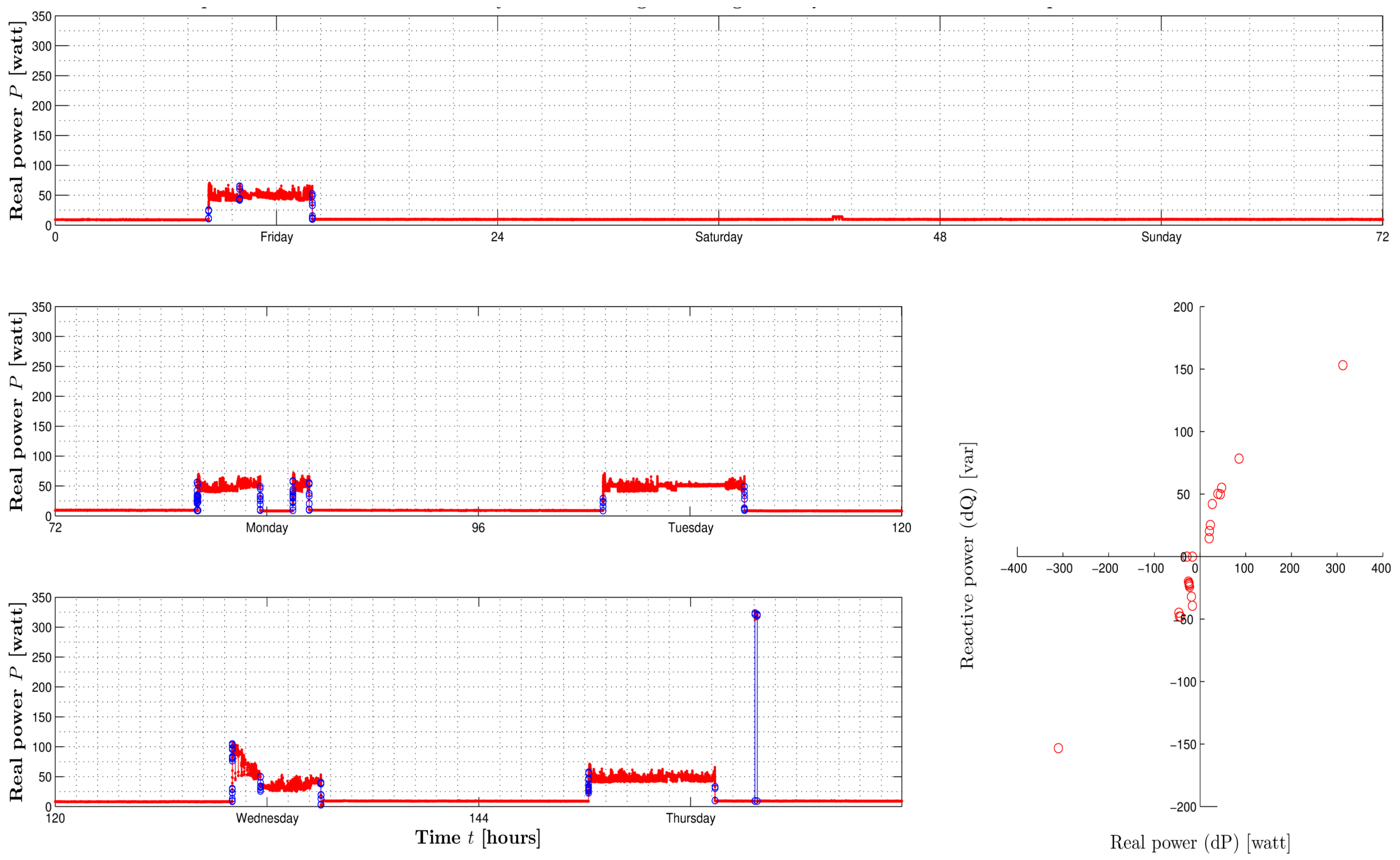


Figure A.24.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

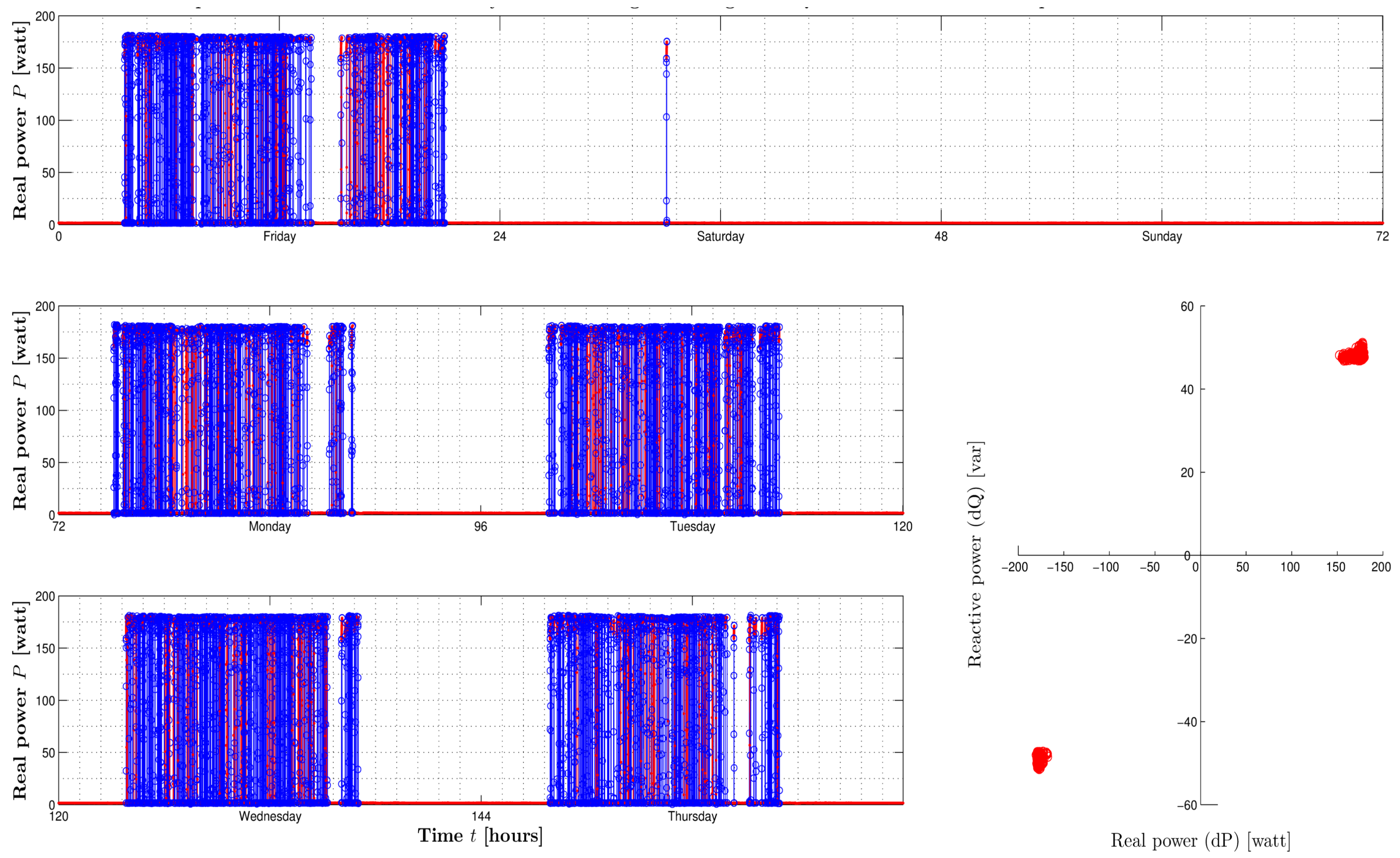


Figure A.25.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

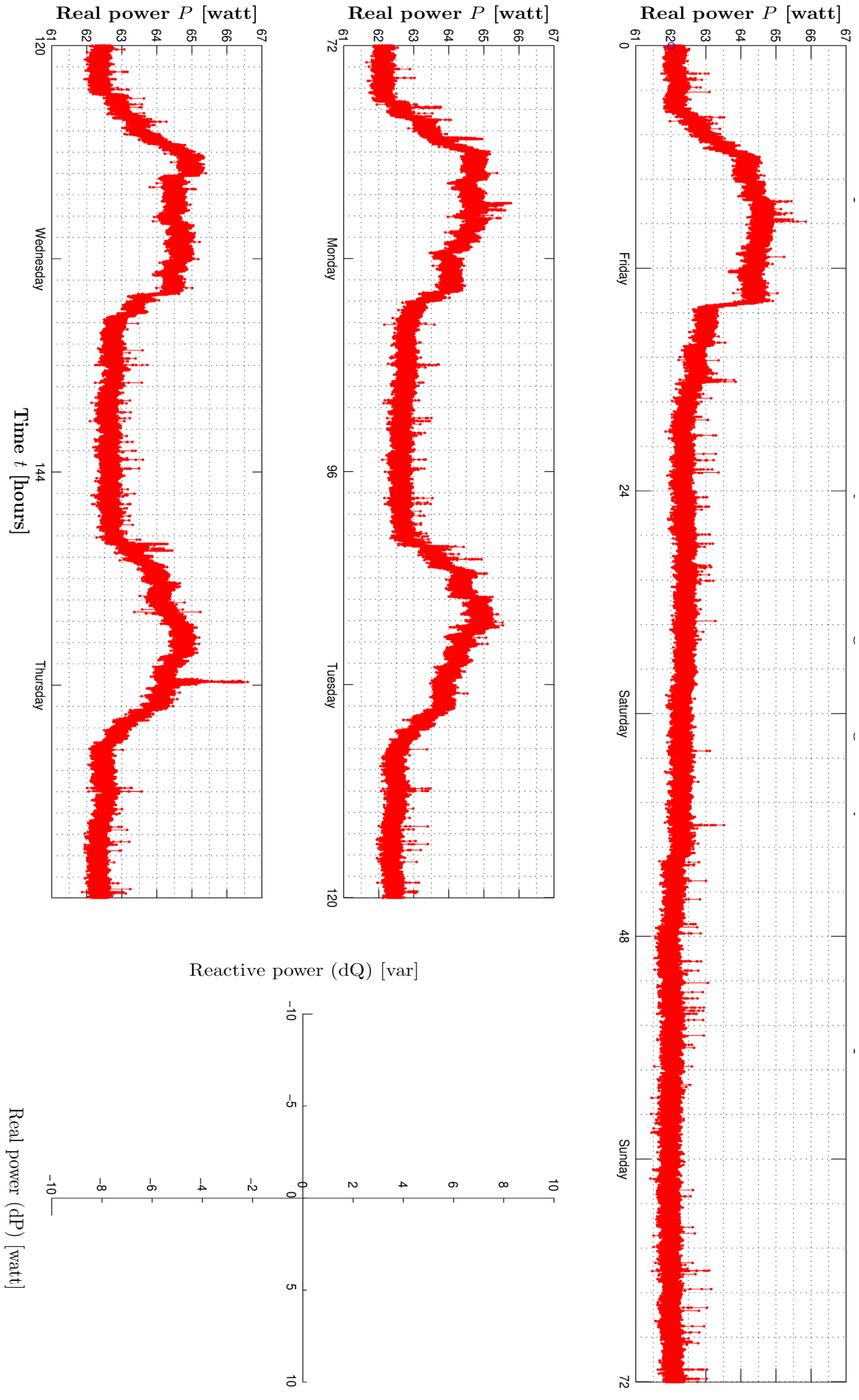


Figure A.26.: Event detection applied to 1 Hz real power signal from commercial building in Berlin.

Appendix B.

BLUED Clustering Validation

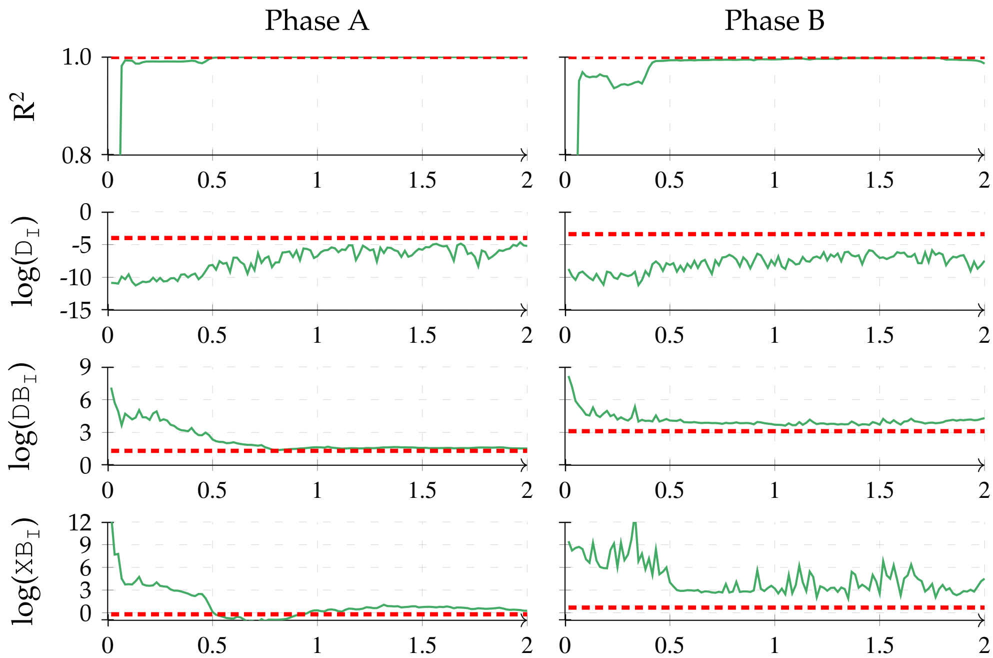


Figure B.1.: Internal cluster validation measures estimated on the two-phase BLUED Dataset. Red line represents the measures of our proposed adaptive segmentation algorithm while green curves are those measured estimated from a fixed window length τ up to 2 seconds.

Appendix C.

Unsupervised Energy Disaggregation on Commercial Data: Results

Figures C.1 to C.3 show the aggregate real power signals of each phase of the commercial dataset leveraged in experimental validation of our unsupervised energy disaggregation framework.

Figures C.4, C.6, C.8 and C.10 show some of the disaggregated load profiles from the first phase of the commercial dataset using the proposed framework, Figure C.12 from the 2nd phase, and Figure C.14 from the 3rd phase.

Best-match sub-metered profiles for each of the disaggregated profiles are depicted in Figures C.5, C.7, C.9, C.11, C.13 and C.15. Note that, sub-metered profiles were acquired using a different power meter that resulted in the calibration artifacts observed from the figures.

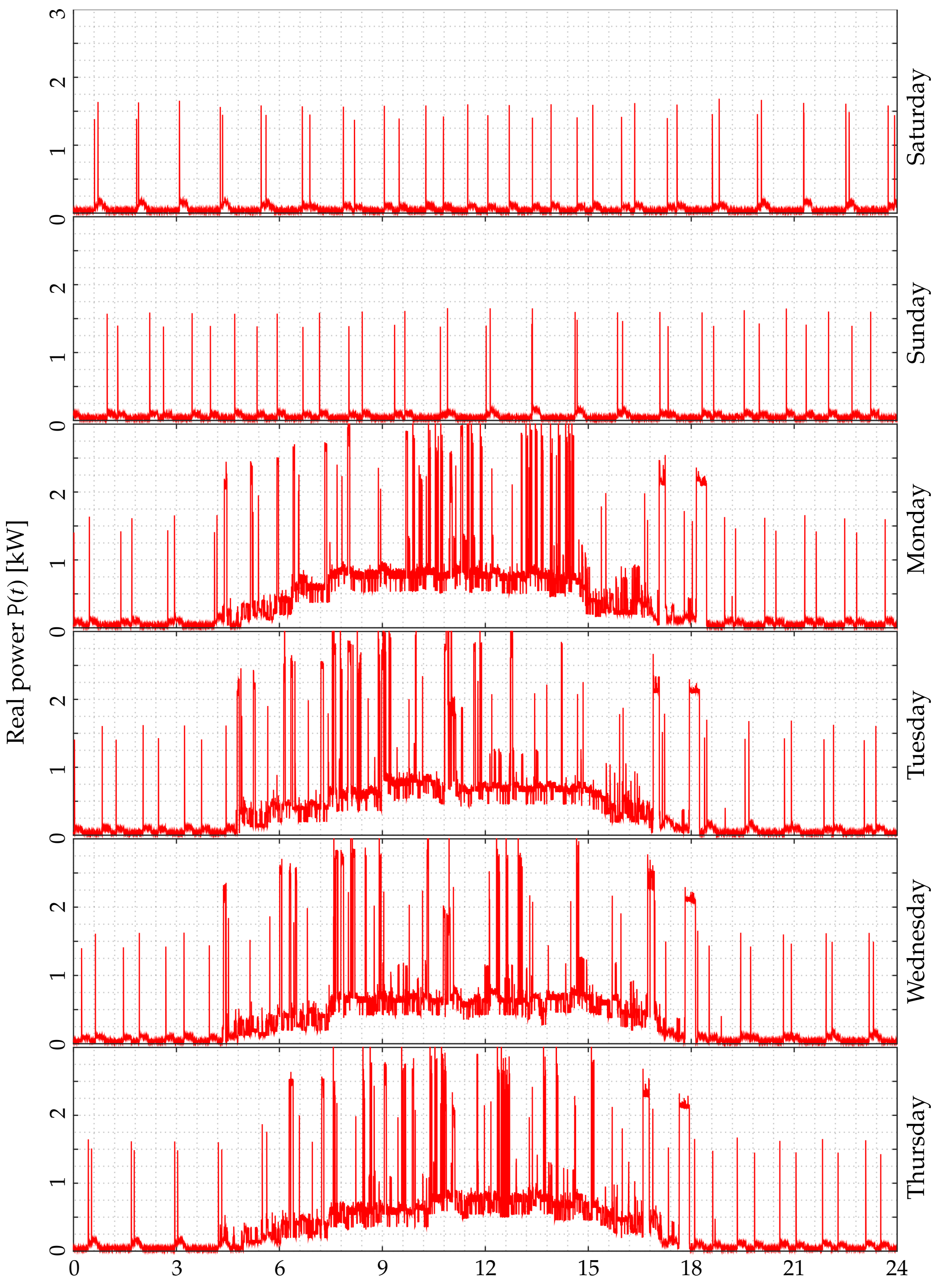


Figure C.1.: Aggregate real power of the 1st phase of a commercial building.

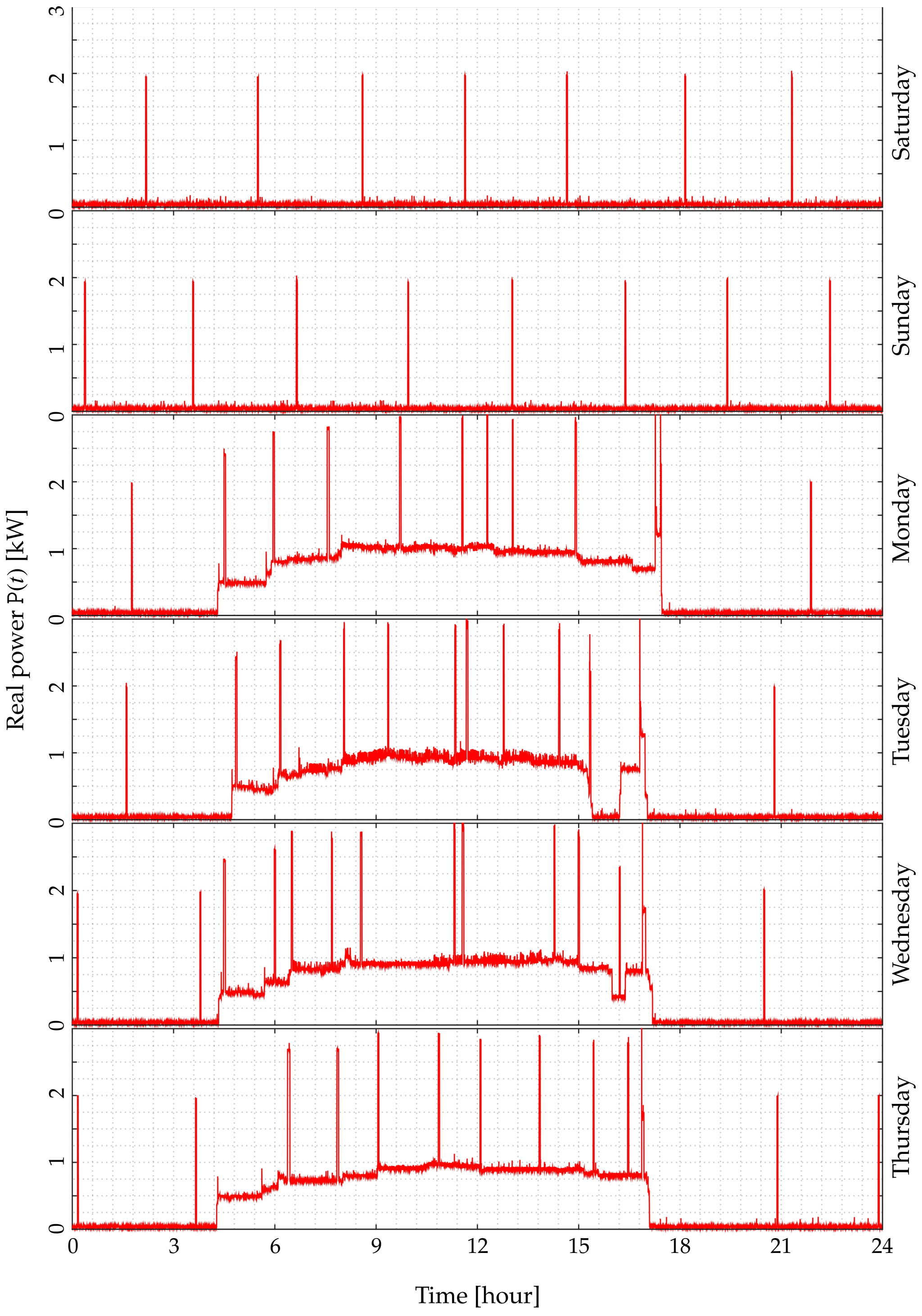


Figure C.2.: Aggregate real power of the 2nd phase of a commercial building.

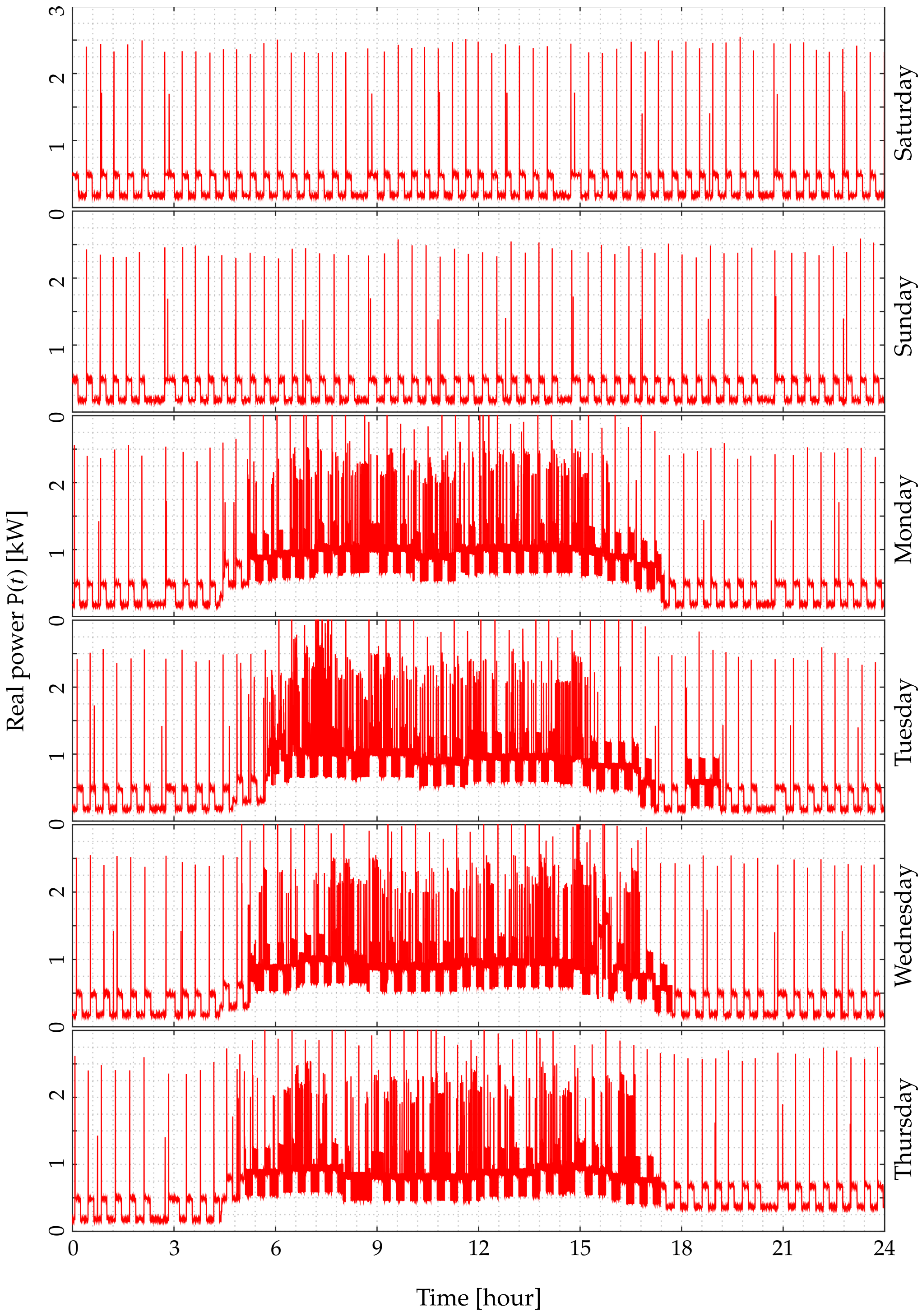


Figure C.3.: Aggregate real power of the 3rd phase of a commercial building.

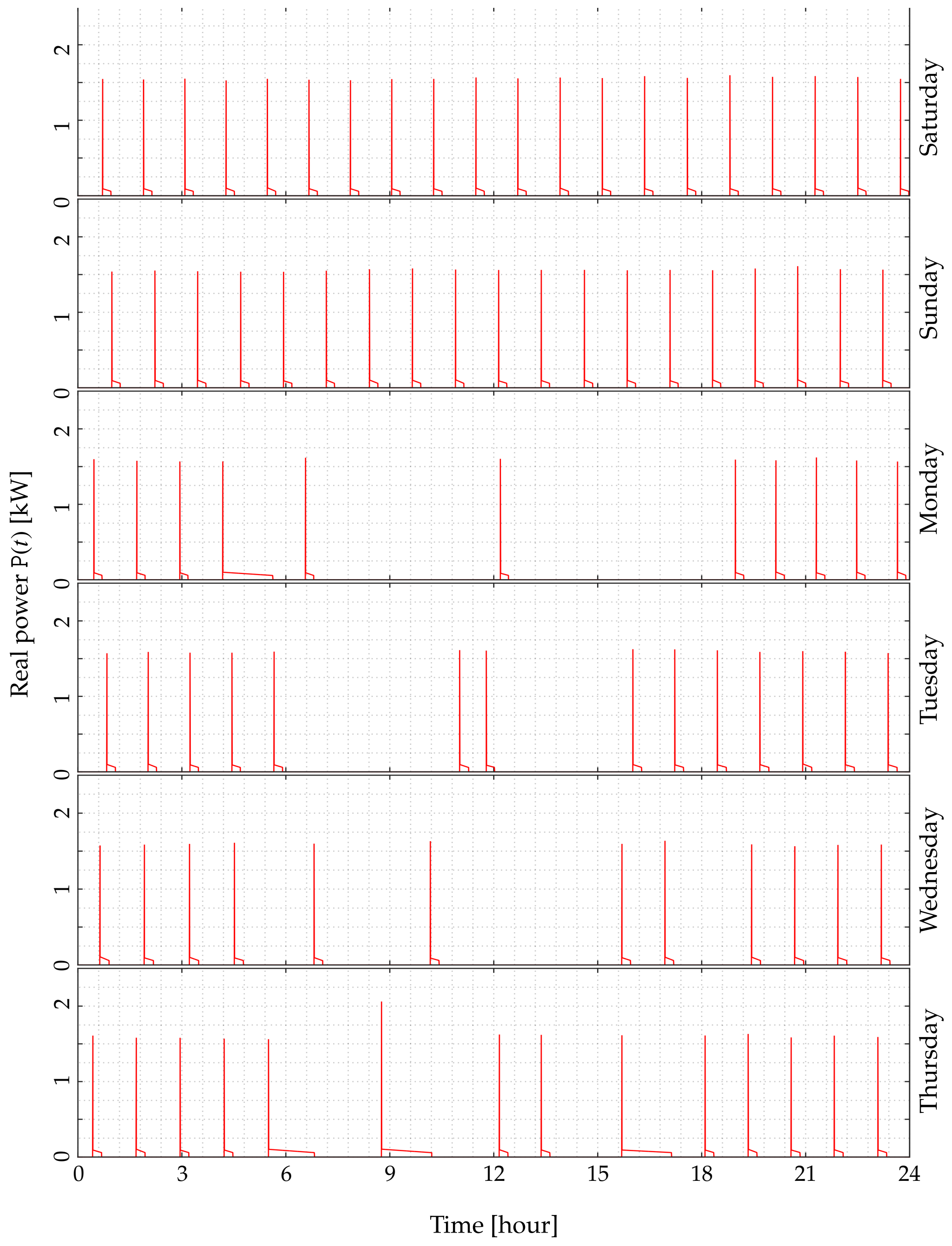


Figure C.4.: Reconstructed profile of a detected load in the 1st phase of the commercial data with estimated energy 2.12 kW·H, 6-day usage time ∼ 27 hours (that is around 18.7% of the disaggregation period), and around 7% of detected events.

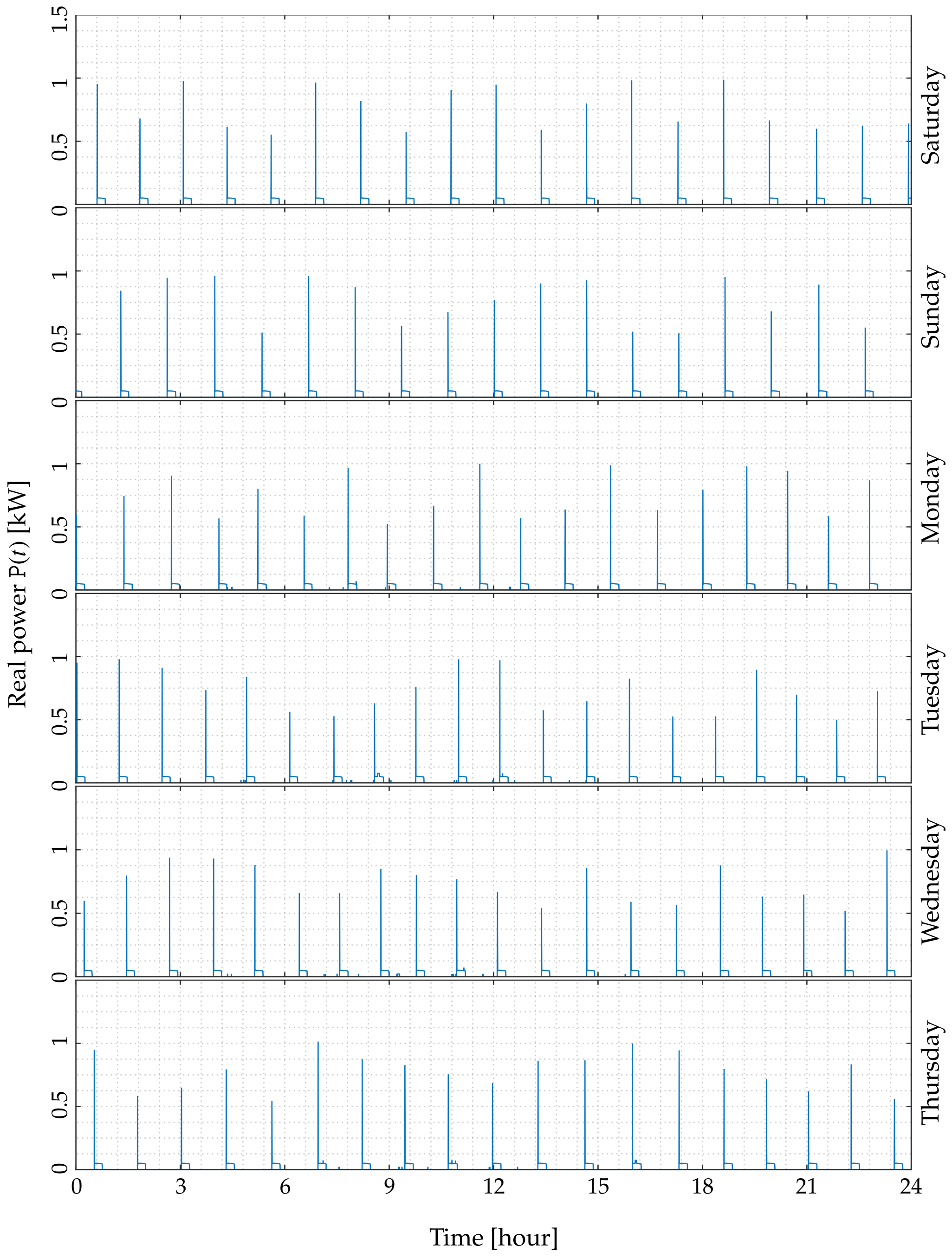


Figure C.5.: Best-match sub-metered load profile to the detected load in Figure C.4. Meter calibration artifacts results in the observed scale difference between the aggregate data meter and the individual load sensors.

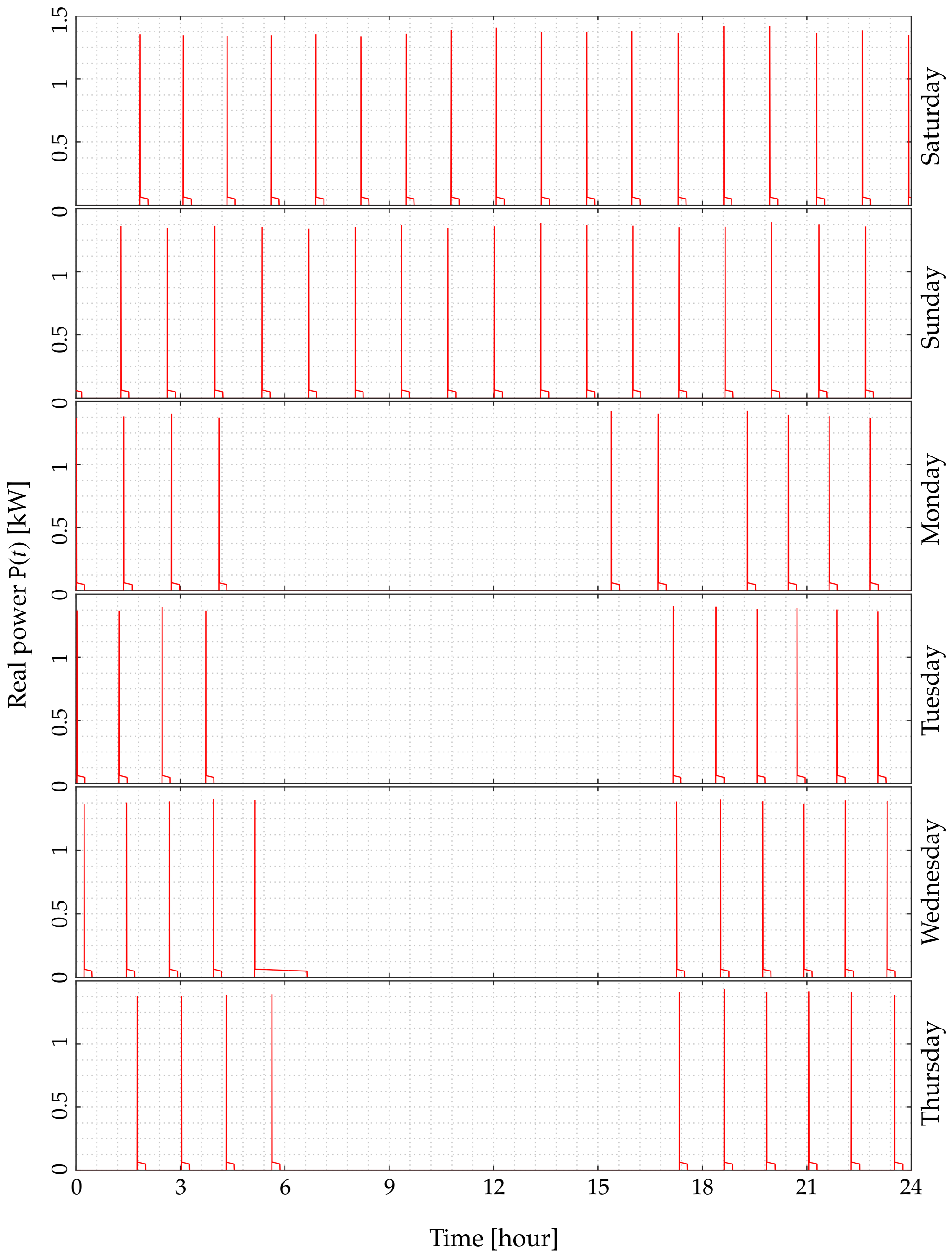


Figure C.6.: Reconstructed profile of a detected load in the 1st phase of the commercial data with estimated energy 1.12 kW·H, 6-day usage time ∼ 19 hours (that is around 13% of the disaggregation period), and around 5.8% of detected events.

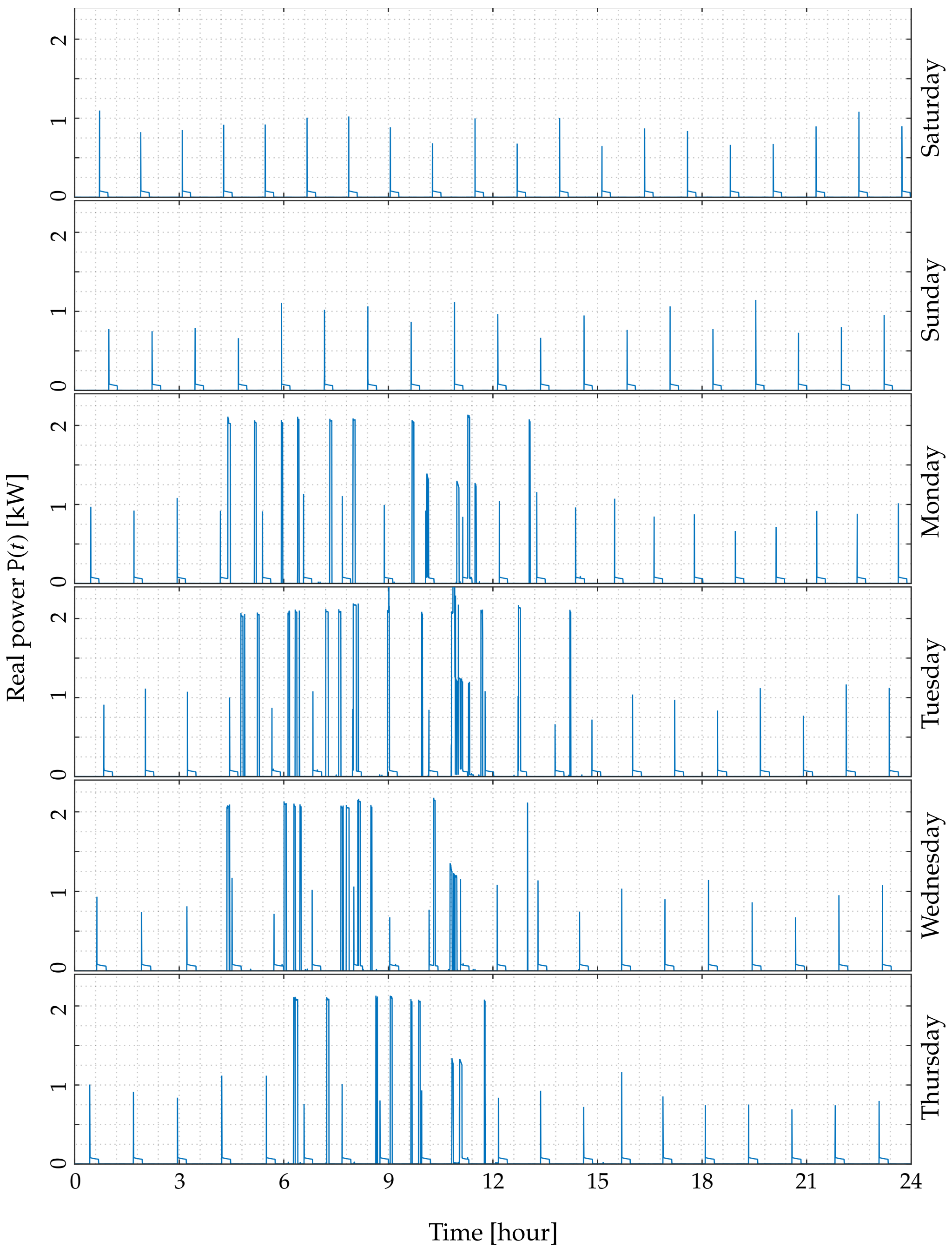


Figure C.7.: Best-match sub-metered load profile to the detected load in Figure C.6. Meter calibration artifacts results in the observed scale difference between the aggregate data meter and the individual load sensors.

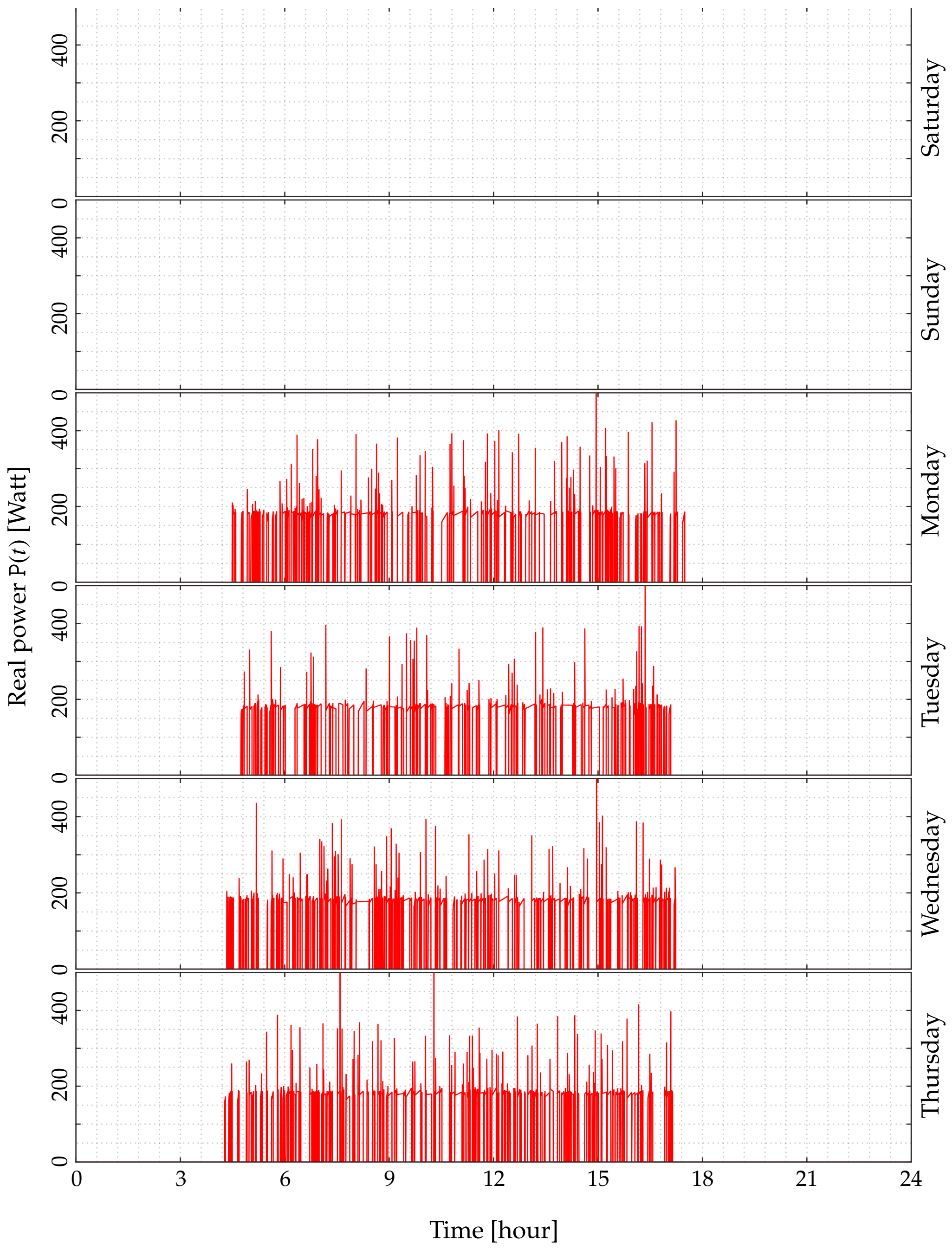


Figure C.8.: Reconstructed profile of a detected load in the 1st phase of the commercial data with estimated energy 5.4 kW·H, 6-day usage time ∼ 30 hours (that is around 21% of the disaggregation period), and around 45% of detected events.

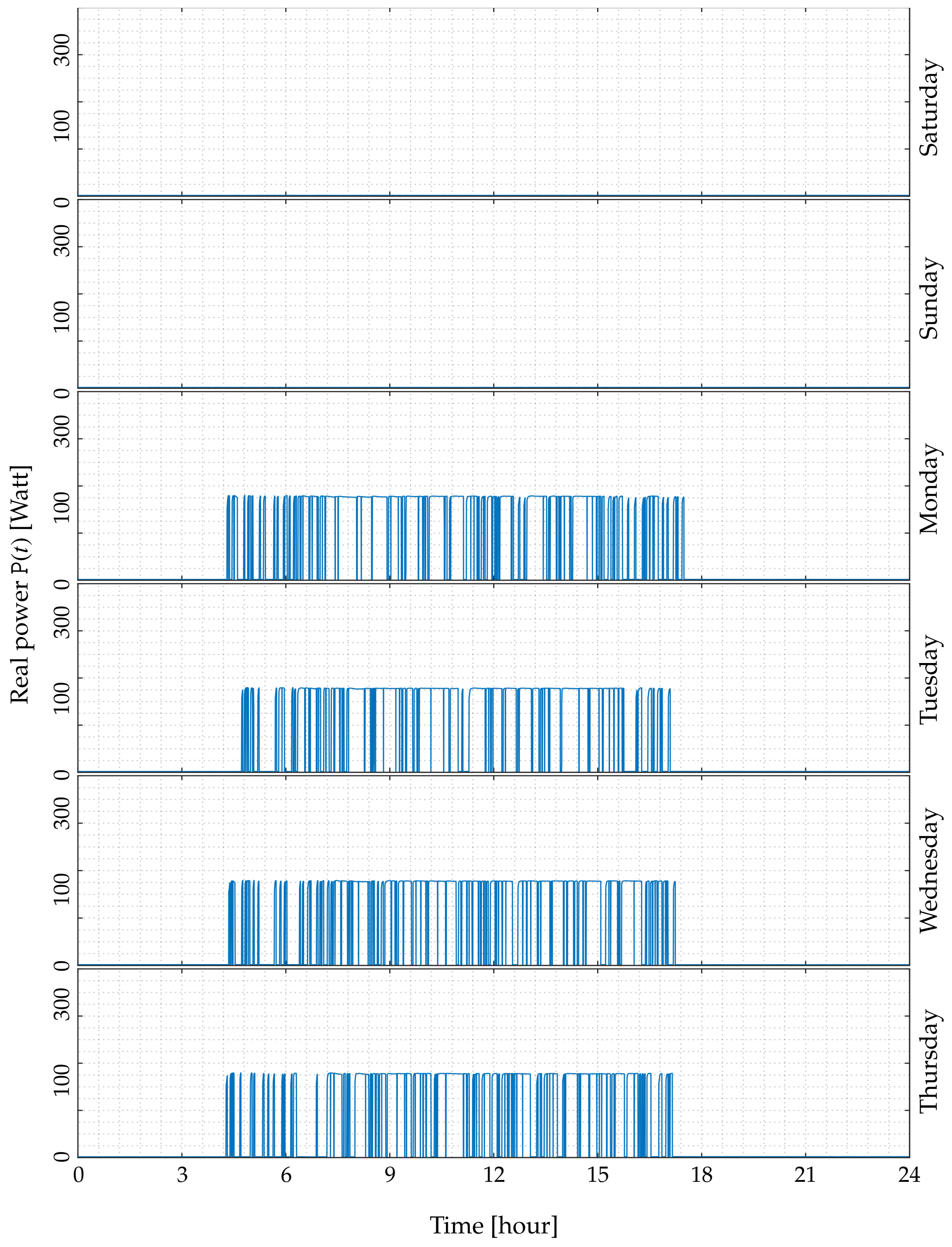


Figure C.9.: Best-match sub-metered load profile to the detected load in Figure C.8. Meter calibration artifacts results in the observed scale difference between the aggregate data meter and the individual load sensors.

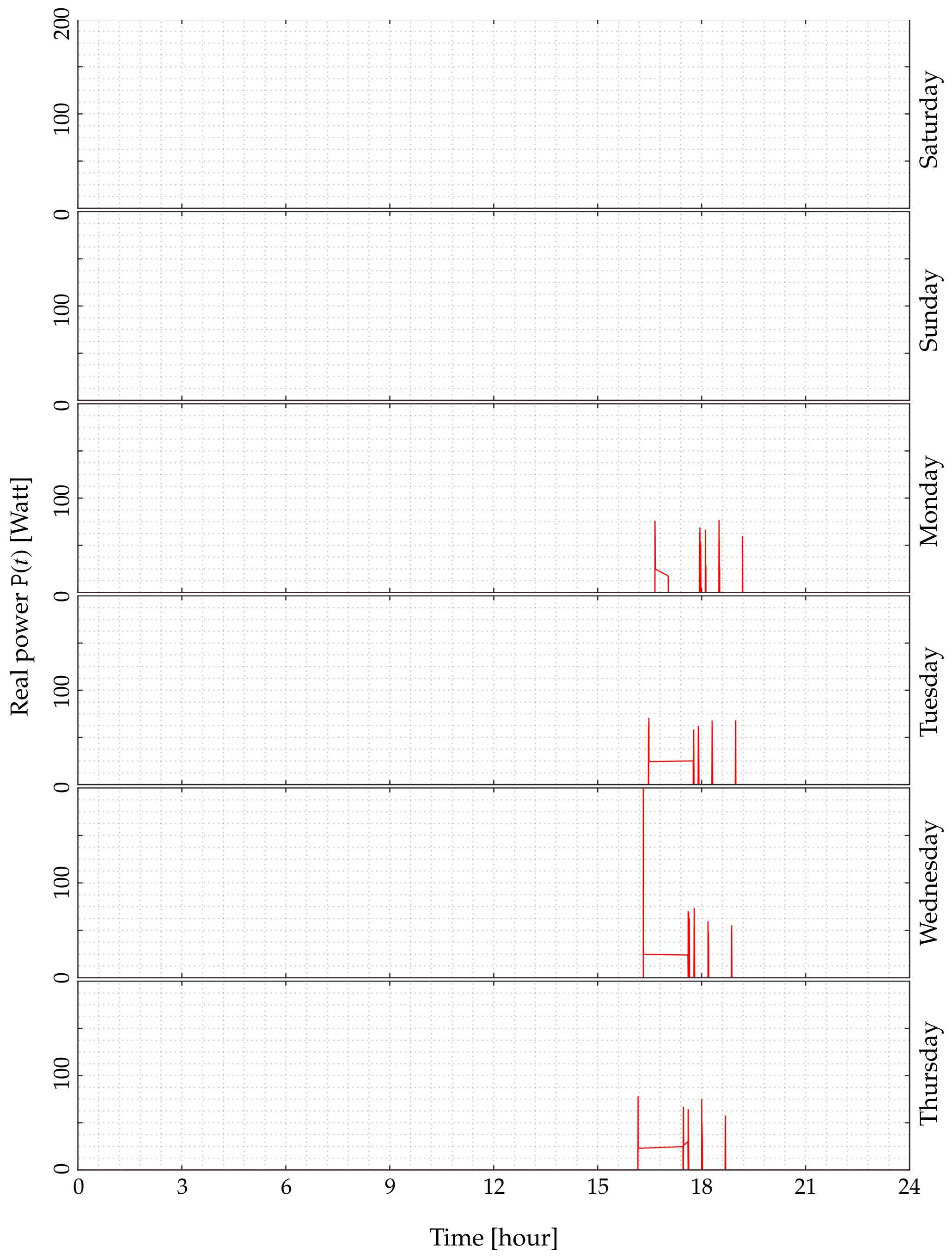


Figure C.10.: Reconstructed profile of a detected load in the 1st phase of the commercial data with estimated energy 0.1 kW·H, 6-day usage time ~ 4.6 hours (that is around 3.2% of the disaggregation period), and around 4.1% of detected events.

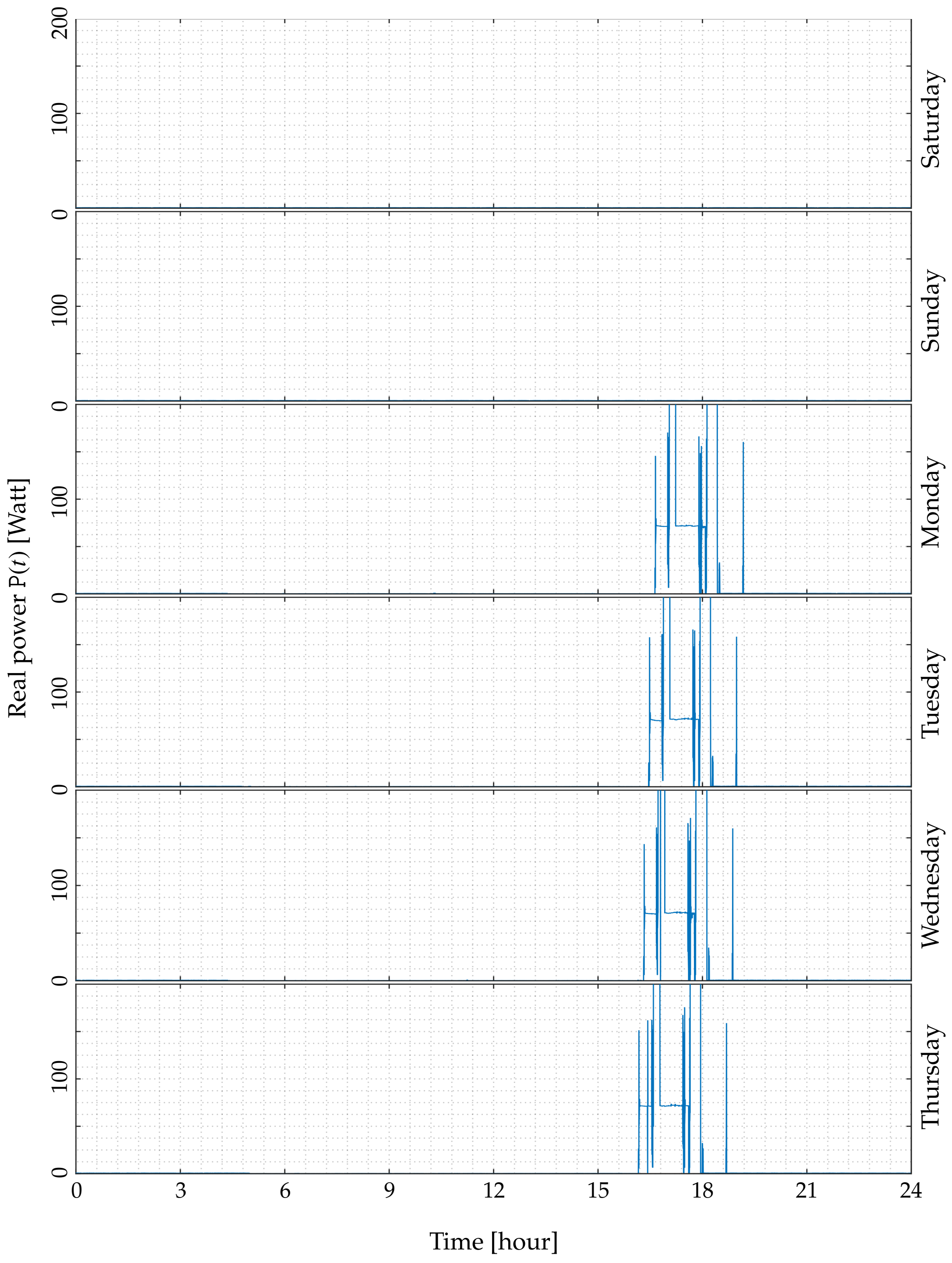


Figure C.11.: Best-match sub-metered load profile to the detected load in Figure C.10. Meter calibration artifacts results in the observed scale difference between the aggregate data meter and the individual load sensors.

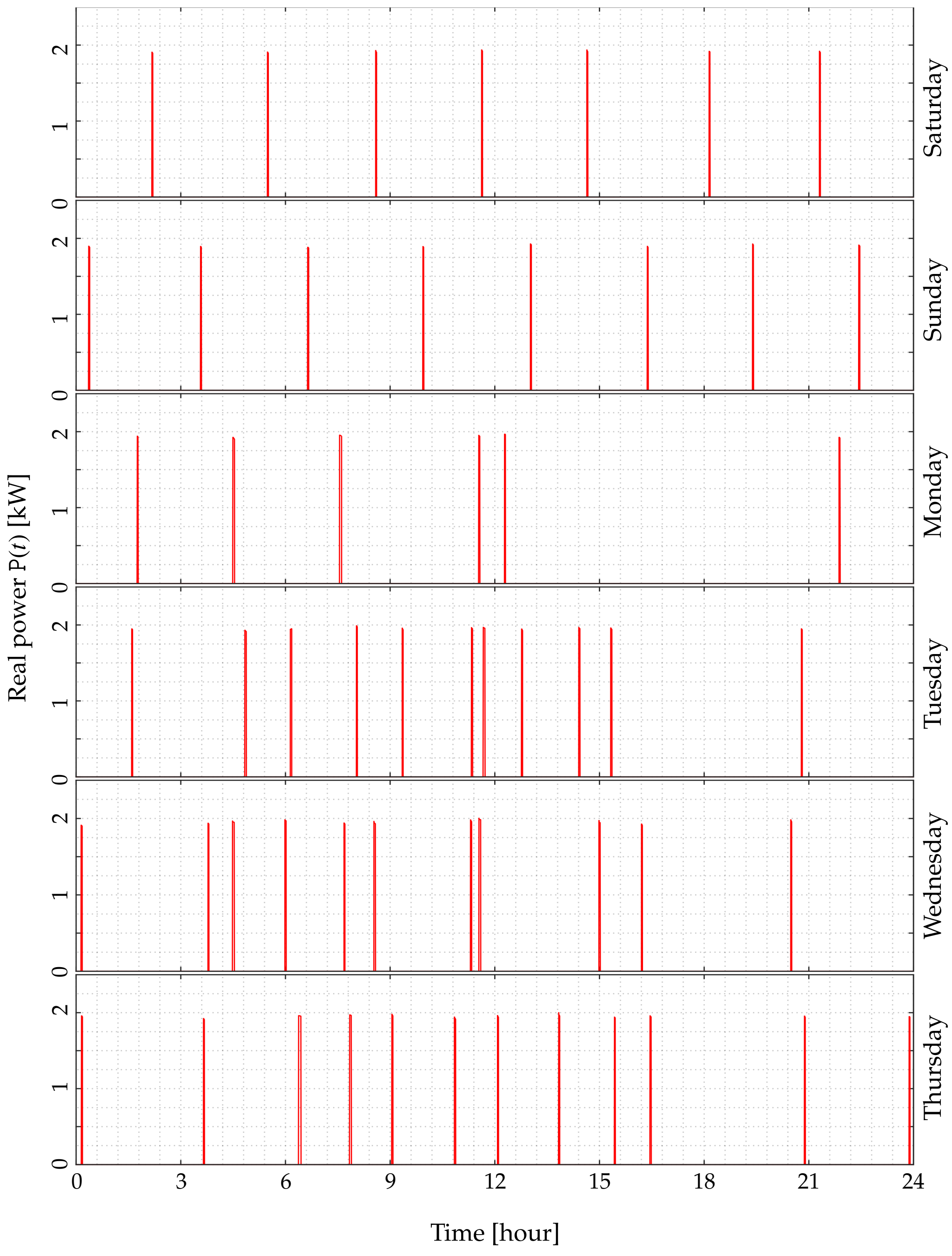


Figure C.12.: Reconstructed profile of a detected load in the 2nd phase of the commercial data with estimated energy 3.5 kW·H, 6-day usage time ~ 1.8 hours (that is around 1.25% of the disaggregation period), and around 60% of detected events.

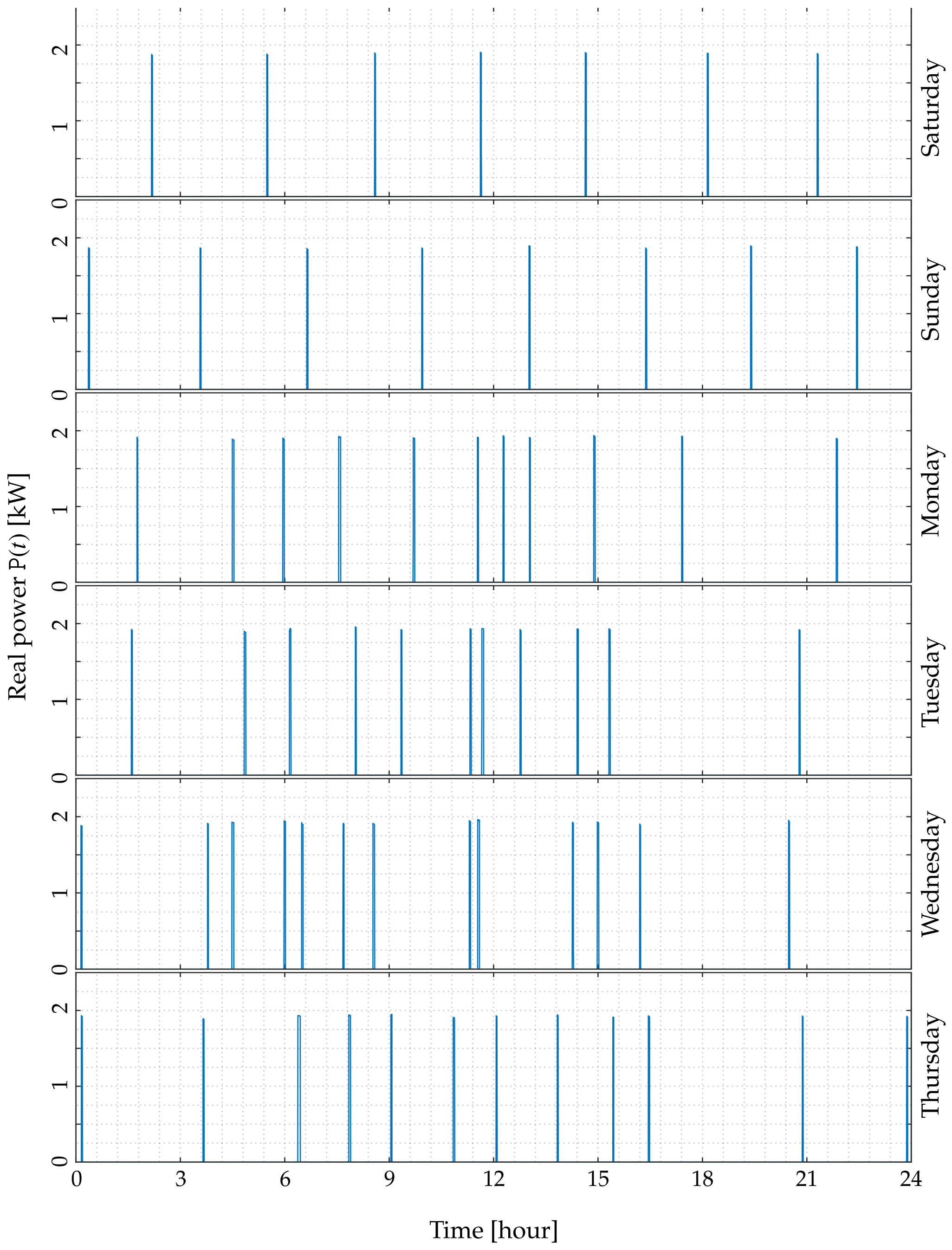


Figure C.13.: Best-match sub-metered load profile to the detected load in Figure C.12. Meter calibration artifacts results in the observed scale difference between the aggregate data meter and the individual load sensors.

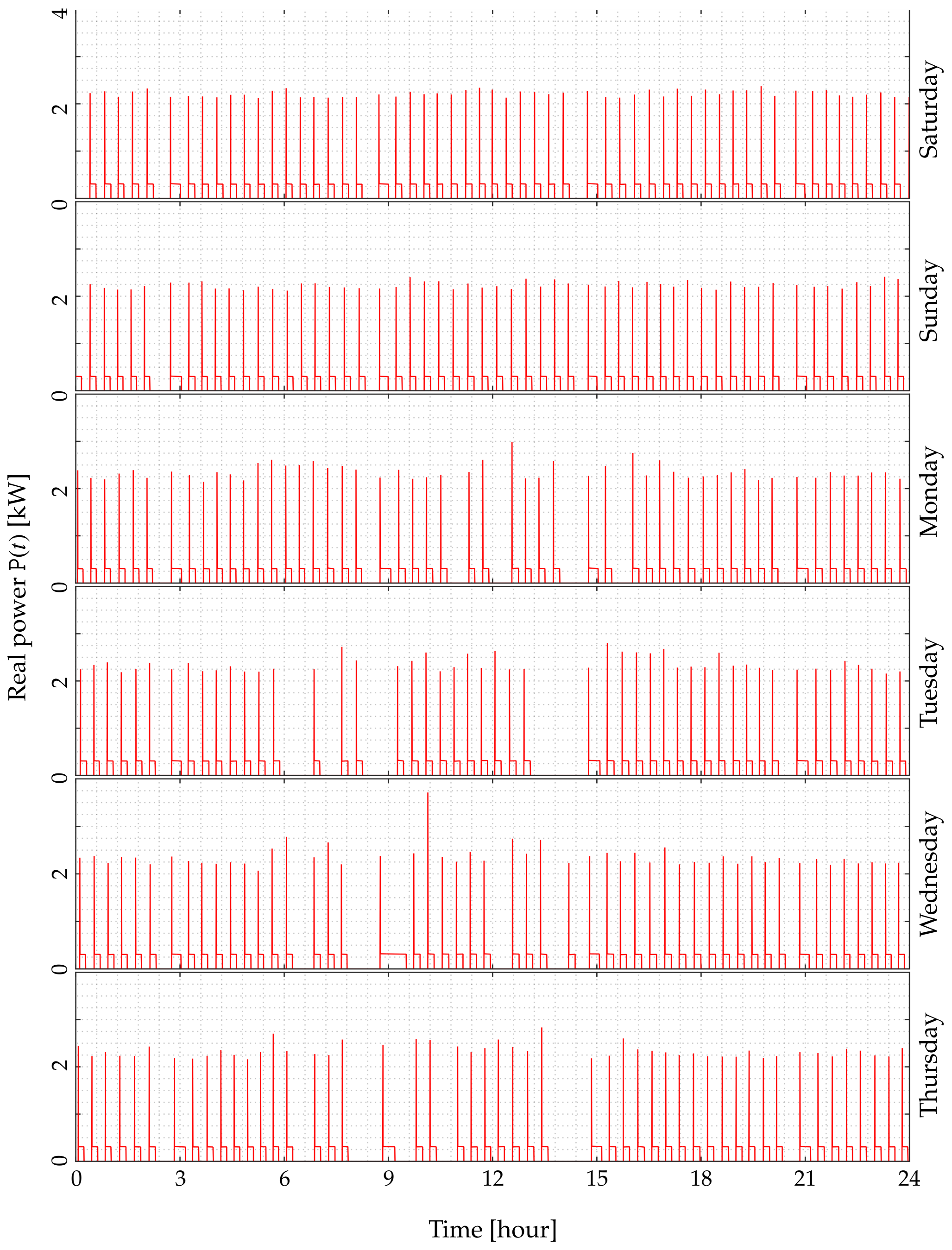


Figure C.14.: Reconstructed profile of a detected load in the 3rd phase of the commercial data with estimated energy 17.8 kW·H, 6-day usage time ∼ 57 hours (that is around 40% of the disaggregation period), and around 52% of detected events.

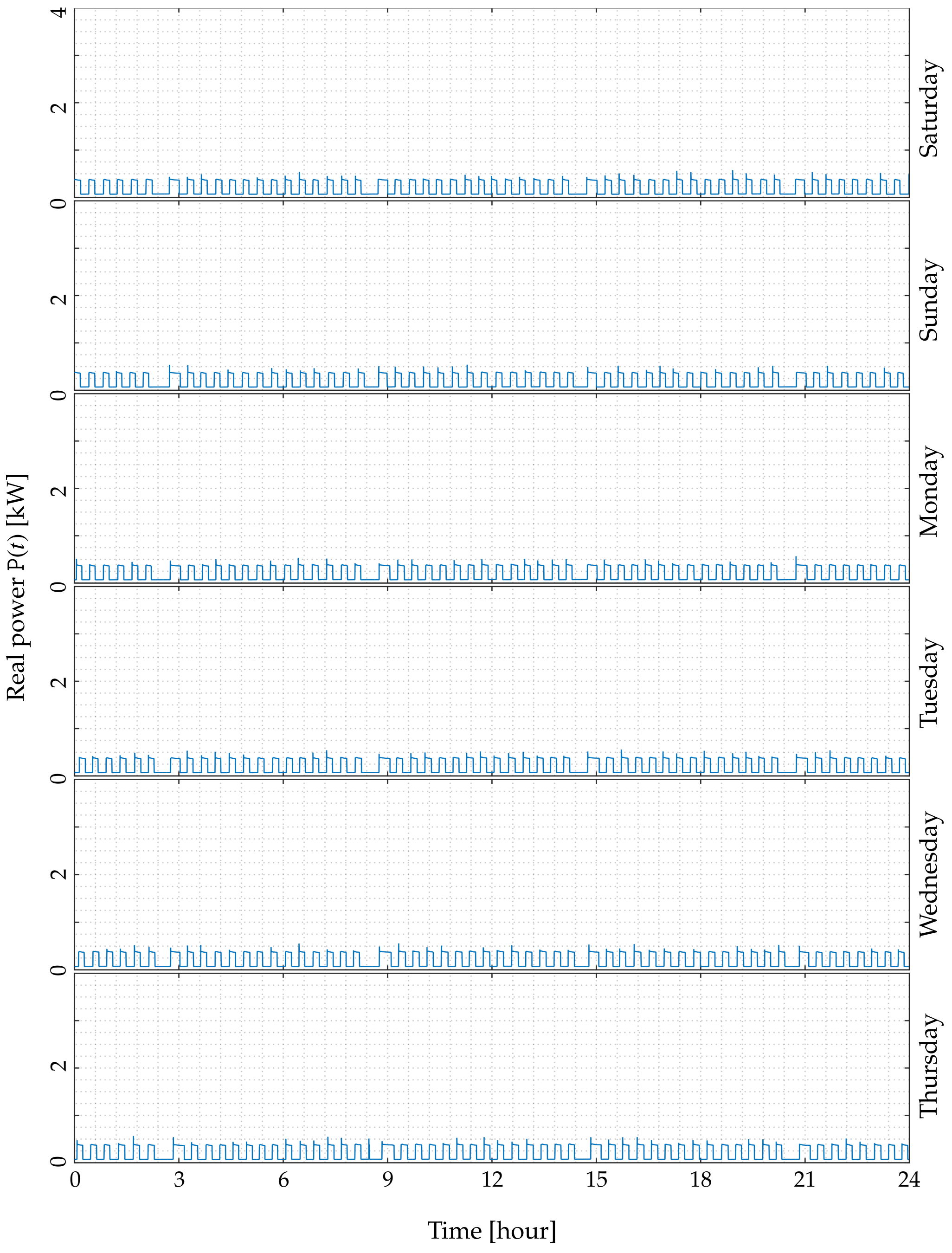


Figure C.15.: Best-match sub-metered load profile to the detected load in Figure C.14. Meter calibration artifacts results in the observed scale difference between the aggregate data meter and the individual load sensors.

Bibliography

Directive 2012/27/EU of the European Parliament and of the Council of the 25th of October 2012 on energy efficiency, Document 32012L0027. *Official Journal of the European Union*, Nov. 2012.

Directive 2012/27/EU of the European Parliament and of the Council of the 25th of October 2012 on energy efficiency, Document 02012L0027-20200101. *Official Journal of the European Union*, Jan. 2020.

C. C. Aggarwal. *Data Classification: Algorithms and Applications*. Chapman & Hall/CRC, 1st edition, 2014. ISBN 1-4665-8674-5 | 978-1-4665-8674-1.

C. C. Aggarwal and C. K. Reddy. *Data Clustering: Algorithms and Applications*. CRC Press, Aug. 2013. ISBN 978-1-4665-5821-2.

A. Agrawala. Learning with a probabilistic teacher. *IEEE Transactions on Information Theory*, 16(4):373–379, July 1970. ISSN 0018-9448, 1557-9654. doi: 10.1109/TIT.1970.1054472.

E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. *Journal of Machine Learning Research (JMLR)*, 1:113–141, Sept. 2001. ISSN 1532-4435. doi: 10.1162/15324430152733133.

M.-R. Amini and P. Gallinari. Semi-supervised Logistic Regression. In *Proceedings of the 15th European Conference on Artificial Intelligence*, ECAI'02, pages 390–394, Amsterdam, The Netherlands, The Netherlands, 2002. IOS Press. ISBN 978-1-58603-257-9.

K. Anderson, A. Ocneanu, D. Benitez, D. Carlson, A. Rowe, and M. Bergés. BLUED: A Fully Labeled Public Dataset for Event-based Non-Intrusive Load Monitoring Research. In *Proceedings of the 2nd KDD Workshop on Data Mining Applications in Sustainability (SustKDD)*, Beijing, China, Aug. 2012.

K. D. Anderson. *Non-Intrusive Load Monitoring: Disaggregation of Energy by Unsupervised Power Consumption Clustering*. Thesis, Carnegie Mellon University, Dec. 2014.

M. Azaza and F. Wallin. Finite State Machine Household's Appliances Models for Non-intrusive Energy Estimation. *Energy Procedia*, 105:2157–2162, May 2017. ISSN 1876-6102. doi: 10.1016/j.egypro.2017.03.609.

L. D. Baets, C. Develder, T. Dhaene, and D. Deschrijver. Automated classification of appliances using elliptical fourier descriptors. In *Proceedings of the IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 153–158, Oct. 2017a. doi: 10.1109/SmartGridComm.2017.8340669.

L. D. Baets, C. Develder, T. Dhaene, D. Deschrijver, J. Gao, and M. Bergés. Handling imbalance in an extended PLAID. In *Proceedings of the Sustainable Internet and ICT for Sustainability (SustainIT)*, pages 1–5, Dec. 2017b. doi: 10.23919/SustainIT.2017.8379795.

S. Bai, J. Z. Kolter, and V. Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv:1803.01271 [cs]*, Apr. 2018.

D. Baptista, S. S. Mostafa, L. Pereira, L. Sousa, and F. Morgado-Dias. Implementation strategy of convolution neural networks on field programmable gate arrays for appliance classification using the voltage and current (v-i) trajectory. *Energies*, 11(9):1–18, 2018.

S. Barker, A. Mishra, D. Irwin, E. Cecchet, and P. Shenoy. Smart*: An Open Data Set and Tools for Enabling Research in Sustainable Homes. In *The 2012 Workshop on Data Mining Applications in Sustainability (SustKDD 2012)*, page 6, Beijing, China, Aug. 2012.

K. S. Barsim. *Unsupervised Non-Intrusive Load Monitoring of Residential Appliances*. Master Thesis, University of Stuttgart, Stuttgart, Germany, July 2013.

K. S. Barsim and B. Yang. Toward a Semi-Supervised Non-Intrusive Load Monitoring System for Event-based Energy Disaggregation. In *Proceedings of the 3rd IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Orlando, Florida, USA, Dec. 2015.

K. S. Barsim and B. Yang. Sequential Clustering-Based Event Detection for Non-Intrusive Load Monitoring. In *Proceedings of the 6th International Conference on Computer Science and Information Technology (CCSIT)*, Zürich, Switzerland, Jan. 2016.

K. S. Barsim and B. Yang. On the Feasibility of Generic Deep Disaggregation for Single-Load Extraction. In *the 4th International Workshop on Non-Intrusive Load Monitoring (NILM)*, Austin, Texas, US, Mar. 2018.

K. S. Barsim, R. Streubel, and B. Yang. Unsupervised Non-Intrusive Load Monitoring of Residential Appliances. In *Proceedings of the 48th International Universities' Power Engineering Conference (UPEC)*, Dublin, Sept. 2013.

K. S. Barsim, R. Streubel, and B. Yang. An Approach for Unsupervised Non-Intrusive Load Monitoring of Residential Appliances. In *the 2nd Non-Intrusive Load Monitoring (NILM) Workshop*, Austin, Texas, USA, Sept. 2014a.

K. S. Barsim, R. Streubel, and B. Yang. Unsupervised Adaptive Event Detection for Building-Level Energy Disaggregation. In *Proceedings of the Power and Energy Student Summit (PESS)*, Stuttgart, Germany, Jan. 2014b.

K. S. Barsim, L. Mauch, and B. Yang. Neural Network Ensembles to Real-time Identification of Plug-level Appliance Measurements. In *Proceedings of the 3rd International Workshop on Non-Intrusive Load Monitoring (NILM)*, Vancouver, Canada, May 2016.

M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes: Theory and Application*. Prentice-Hall, Inc., USA, 1993. ISBN 978-0-13-126780-0.

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 (ACM e-Energy)*, Cambridge, UK, 2014a.

N. Batra, O. Parson, M. Berges, A. Singh, and A. Rogers. A Comparison of Non-Intrusive Load Monitoring Methods for Commercial and Residential Buildings. *arXiv preprint, arXiv:1408.6595*, Aug. 2014b.

N. Batra, A. Singh, and K. Whitehouse. Neighbourhood NILM: A Big-data Approach to Household Energy Disaggregation. *CoRR*, abs/1511.02900, 2015.

A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. Automatic differentiation in machine learning: A survey. *arXiv:1502.05767 [cs, stat]*, Feb. 2018.

E. Bayer. Report on the German Power System. Technical Report 1.2, Agora Energiewende, Berlin, Germany, Feb. 2015.

J. Bégaint, F. Galpin, P. Guillotel, and C. Guillemot. Deep Frame Interpolation for Video Compression. In *2019 Data Compression Conference (DCC)*, pages 556–556, Mar. 2019. doi: 10.1109/DCC.2019.00068.

BEIS. Smart Metering Implementation Programme: Progress Report for 2018. Technical report, Department for Business, Energy, and Industrial Strategy, Dec. 2018.

Belkin. *Belkin Energy Disaggregation Competition*. https://www.kaggle.com/, Oct. 2013.

A. Ben-David. Comparison of Classification Accuracy Using Cohen's Weighted Kappa. *Expert Systems with Applications*, 34(2):825–832, Feb. 2008. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2006.10.022.

Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, Aug. 2013. doi: 10.1109/TPAMI.2013.50.

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0-387-31073-8.

A. Blum and T. Mitchell. Combining Labeled and Unlabeled Data with Co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, pages 92–100, New York, NY, USA, 1998. ACM. ISBN 978-1-58113-057-7. doi: 10.1145/279943.279962.

R. Bonfigli, S. Squartini, M. Fagiani, and F. Piazza. Unsupervised algorithms for non-intrusive load monitoring: An up-to-date overview. In *2015 IEEE 15th International Conference on Environment and Electrical Engineering (EEEIC)*, pages 1175–1180, Rome, Italy, June 2015. IEEE. ISBN 978-1-4799-7993-6. doi: 10.1109/EEEIC.2015.7165334.

L. Bottou. Stochastic Gradient Descent Tricks. In G. Montavon, G. B. Orr, and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade: Second Edition*, pages 421–436. Springer, Berlin, Heidelberg, 2012. ISBN 978-3-642-35289-8. doi: 10.1007/978-3-642-35289-8_25.

BP. BP Statistical Review of World Energy 68$^{\text{th}}$ Ed. Technical Report 68, BP, London, UK, June 2019.

L. Breiman. Bagging Predictors. *Machine Learning*, 24(2):123–140, Aug. 1996. ISSN 1573-0565. doi: 10.1023/A:1018054314350.

J. C. Caicedo and S. Lazebnik. Active Object Localization with Deep Reinforcement Learning. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2488–2496, Dec. 2015. doi: 10.1109/ICCV.2015.286.

K. Carrie Armel, A. Gupta, G. Shrimali, and A. Albert. Is disaggregation the holy grail of energy efficiency? The case of electricity. *Energy Policy*, 52:213–234, Jan. 2013. ISSN 0301-4215. doi: 10.1016/j.enpol.2012.08.062.

H.-H. Chang, H.-T. Yang, and C.-L. Lin. Load Identification in Neural Networks for a Non-intrusive Monitoring of Industrial Electrical Loads. In W. Shen, J. Yong, Y. Yang, J.-P. A. Barthès, and J. Luo, editors, *Computer Supported Cooperative Work in Design IV*, Lecture Notes in Computer Science, pages 664–674, Berlin, Heidelberg, 2008. Springer. ISBN 978-3-540-92719-8. doi: 10.1007/978-3-540-92719-8_60.

H.-H. Chang, C.-L. Lin, and J.-K. Lee. Load Identification in Nonintrusive Load Monitoring using Steady-state and Turn-on Transient Energy Algorithms. In *In Proceedings of the 14\textsuperscriptth International Conference on Computer Supported Cooperative Work in Design*, pages 27–32, Apr. 2010. doi: 10.1109/CSCWD.2010.5472008.

O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass, 2006. ISBN 978-0-262-03358-9. OCLC: ocm64898359.

D. Chen, D. Irwin, and P. Shenoy. SmartSim: A device-accurate smart home simulator for energy analytics. In *2016 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 686–692, Nov. 2016. doi: 10.1109/SmartGridComm.2016.7778841.

K.-L. Chen, H.-H. Chang, and N. Chen. A new transient feature extraction method of power signatures for Nonintrusive Load Monitoring Systems. In *2013 IEEE International Workshop on Applied Measurements for Power Systems (AMPS)*, pages 79–84, Sept. 2013. doi: 10.1109/AMPS.2013.6656230.

R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 6571–6583. Curran Associates, Inc., 2018.

B. Christian, W. Kleiminger, R. Cicchetti, T. Staake, and S. Santini. The ECO Data Set and the Performance of Non-Intrusive Load Monitoring Algorithms. In *Proceedings of the 1st ACM International Conference on Embedded Systems for Energy-Efficient Buildings (BuildSys)*, pages 80–89, Memphis, TN, USA, Nov. 2014. ACM.

D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649, Providence, RI, June 2012. IEEE. ISBN 978-1-4673-1228-8 978-1-4673-1226-4 978-1-4673-1227-1. doi: 10.1109/CVPR.2012.6248110.

J. Cohen. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46, Apr. 1960. ISSN 0013-1644. doi: 10.1177/001316446002000104.

A. Cole and A. Albicki. Data extraction for effective non-intrusive identification of residential power loads. In *IMTC/98 Conference Proceedings. IEEE Instrumentation and Measurement Technology Conference. Where Instrumentation Is Going (Cat. No.98CH36222)*, volume 2, pages 812–815 vol.2, May 1998. doi: 10.1109/IMTC.1998.676838.

M. Collins and Y. Singer. Unsupervised Models for Named Entity Classification. In *In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 100–110, 1999.

A. Cominola, M. Giuliani, D. Piga, A. Castelletti, and A. E. Rizzoli. A Hybrid Signature-based Iterative Disaggregation algorithm for Non-Intrusive Load Monitoring. *Applied Energy*, 185(P1):331–344, 2017. ISSN 0306-2619. doi: 10.1016/j.apenergy.2016.10.040.

A. Cooper and M. Shuster. Electric Company Smart Meter Deployments: Foundation for a Smart Grid (2019 Update). Technical report, Institute for Electric Innovation, The Edison Foundation, Dec. 2019.

D. L. Davies and D. W. Bouldin. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, Apr. 1979. ISSN 1939-3539. doi: 10.1109/TPAMI.1979.4766909. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

P. Davies, J. Dennis, J. Hansom, W. Martin, A. Stankevicius, and L. Ward. Deep Neural Networks for Appliance Transient Classification. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8320–8324, Brighton, United Kingdom, May 2019. ISBN 978-1-4799-8131-1. doi: 10.1109/ICASSP.2019.8682658.

L. De Baets, T. Dhaene, D. Deschrijver, C. Develder, and M. Berges. VI-Based Appliance Classification Using Aggregated Power Consumption Data. In *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 179–186, Taormina, June 2018. IEEE. ISBN 978-1-5386-4705-9. doi: 10.1109/SMARTCOMP.2018.00089.

M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3844–3852. Curran Associates, Inc., 2016.

B. der Energie-und Wasserwirtschaft (BDEW). Energy Market Germany. https://www.bdew.de/service/daten-und-grafiken/nettostromverbrauch-nach-verbrauchergruppen/, Aug. 2019.

J. Desmedt, G. Vekemans, and D. Maes. Ensuring effectiveness of information to influence household behaviour. *Journal of Cleaner Production*, 17(4):455–462, Mar. 2009. ISSN 0959-6526. doi: 10.1016/j.jclepro.2008.08.017.

T. Dozat. Incorporating Nesterov Momentum into Adam. Technical Report 54, Standford University, May 2015.

S. Drenker and A. Kader. Nonintrusive monitoring of electric loads. *IEEE Computer Applications in Power*, 12(4):47–51, Oct. 1999. ISSN 1558-4151. doi: 10.1109/67.795138.

L. Duncker, G. Bohner, J. Boussard, and M. Sahani. Learning interpretable continuous-time models of latent stochastic dynamical systems. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1726–1734, Long Beach, California, USA, June 2019. PMLR.

J. C. Dunn. Well-Separated Clusters and Optimal Fuzzy Partitions. *Journal of Cybernetics*, 4(1):95–104, Jan. 1974. ISSN 0022-0280. doi: 10.1080/01969727408546059. _eprint: https://doi.org/10.1080/01969727408546059.

G. Ernst & Young. Cost-benefit analysis for the comprehensive use of smart metering. Technical report, Ernst \& Yound on behalf of the Federal Ministry of Economics and Technology, 2013.

M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, Jan. 1996.

A. M. Fatouh and O. A. Nasr. New Semi-Supervised and Active Learning Combination Technique for Non-Intrusive Load Monitoring. In *2018 IEEE International Conference on Smart Energy Grid Engineering (SEGE)*, pages 181–185, Aug. 2018. doi: 10.1109/SEGE.2018.8499498.

F. M. for Economic Affairs and Energy. Gesetz zur Digitalisierung der Energiewende: Bundesgesetzblatt Jahrgang 2016 Teil I Nr. 43, a.

F. M. for Economic Affairs and Energy. Variantenrechnungen von in Diskussion befindlichen Rollout-Strategien - Ergänzungen zur KNA vom Juli 2013. Technical report, b.

F. M. for Economic Affairs and Energy (BMWi). The Energy of the Future: Second Progress Report on the Energy Transition (Reporting Year 2016). Technical report, Federal Ministry for Economic Affairs and Energy (BMWi), Munich, Germany, June 2018.

F. M. for Economic Affairs and Energy (BMWi). The Energy of the Future: Second Progress Report on the Energy Transition (Reporting Year 2017). Technical report, Federal Ministry for Economic Affairs and Energy (BMWi), Munich, Germany, June 2019.

S. Fralick. Learning to recognize patterns without a teacher. *IEEE Transactions on Information Theory*, 13(1):57–64, Jan. 1967. ISSN 0018-9448, 1557-9654. doi: 10. 1109/TIT.1967.1053952.

J. H. Friedman and P. Hall. On bagging and nonlinear estimation. *Journal of Statistical Planning and Inference*, 137(3):669–683, Mar. 2007. ISSN 0378-3758. doi: 10.1016/j.jspi.2006.06.002.

J. Froehlich, E. Larson, S. Gupta, G. Cohn, M. Reynolds, and S. Patel. Disaggregated End-Use Energy Sensing for the Smart Grid. *IEEE Pervasive Computing*, 10(1): 28–39, Jan. 2011. ISSN 1558-2590. doi: 10.1109/MPRV.2010.74.

K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, Jan. 1975. ISSN 1557-9654. doi: 10.1109/TIT.1975.1055330. Conference Name: IEEE Transactions on Information Theory.

Y. Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, Sept. 2016.

J. Gao, S. Giri, E. C. Kara, and M. Bergés. PLAID: A Public Dataset of High-resolution Electrical Appliance Measurements for Load Identification Research: Demo Abstract. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, BuildSys '14, pages 198–199, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3144-9. doi: 10.1145/2674061.2675032.

J. Gao, E. C. Kara, S. Giri, and M. Bergés. A Feasibility Study of Automated Plug-Load Identification from High-Frequency Measurements. In *In Proceedings of the 3rd IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 220–224, Orlando, FL, USA, Dec. 2015. ISBN 978-1-4799-7591-4. doi: 10.1109/GlobalSIP.2015.7418189.

J. Gast and S. Roth. Deep Video Deblurring: The Devil is in the Details. *arXiv:1909.12196 [cs]*, Sept. 2019.

J. Gillis and W. G. Morsi. Non-intrusive load monitoring using orthogonal wavelet analysis. In *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–5, May 2016. doi: 10.1109/CCECE.2016.7726786.

J. M. Gillis and W. G. Morsi. Non-Intrusive Load Monitoring Using Semi-Supervised Machine Learning and Wavelet Design. *IEEE Transactions on Smart Grid*, 8:2648–2655, 2017. doi: 10.1109/tsg.2016.2532885.

S. Giri and M. Bergés. An energy estimation framework for event-based methods in Non-Intrusive Load Monitoring. *Energy Conversion and Management*, 90:488–498, Jan. 2015. ISSN 0196-8904. doi: 10.1016/j.enconman.2014.11.047.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh and M. Titterington, editors, *Proceedings of the 13th*

International Conference on Artificial Intelligence and Statistics, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, May 2010. PMLR.

A. B. Goldberg. *New Directions in Semi-Supervised Learning*. PhD Thesis, University of Wisconsin at Madison, Madison, WI, USA, 2010.

M. Goyal, K. Tatwawadi, S. Chandak, and I. Ochoa. DeepZip: Lossless Data Compression Using Recurrent Neural Networks. In *2019 Data Compression Conference (DCC)*, pages 575–575, Mar. 2019. doi: 10.1109/DCC.2019.00087.

Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew. Deep learning for visual understanding: A review. *Neurocomputing*, 187:27–48, Apr. 2016. ISSN 0925-2312. doi: 10.1016/j.neucom.2015.09.116.

S. Gupta, M. S. Reynolds, and S. N. Patel. ElectriSense: Single-Point Sensing Using EMI for Electrical Event Detection and Classification in the Home. In *In Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, pages 139–148, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-843-8. doi: 10.1145/1864349.1864375.

M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On Clustering Validation Techniques. *Journal of Intelligent Information Systems*, 17(2):107–145, Dec. 2001. ISSN 1573-7675. doi: 10.1023/A:1012801612483.

G. W. Hart. Prototype Non-Intrusive Appliance Load Monitor. Progress Report 2, MIT Energy Laboratory and Electric Power Research Institute, Concord, Massachusetts 01742, Sept. 1985.

G. W. Hart. Residential Energy Monitoring and Computerized Surveillance via Utility Power Flows. *IEEE Technology and Society Magazine*, 8(2):12–16, June 1989. ISSN 0278-0097. doi: 10.1109/44.31557.

G. W. Hart. Nonintrusive Appliance Load Monitoring. *Proceedings of the IEEE*, 80 (12):1870–1891, Dec. 1992. ISSN 0018-9219. doi: 10.1109/5.192069.

T. Hassan, F. Javed, and N. Arshad. An Empirical Investigation of V-I Trajectory Based Load Signatures for Non-Intrusive Load Monitoring. *IEEE Transactions on Smart Grid*, 5(2):870–878, Mar. 2014. doi: 10.1109/TSG.2013.2271282.

M. Hazas, A. Friday, and J. Scott. Look Back before Leaping Forward: Four Decades of Domestic Energy Inquiry. *IEEE Pervasive Computing*, 10(1):13–19, Jan. 2011. ISSN 1558-2590. doi: 10.1109/MPRV.2010.89.

K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016a. doi: 10.1109/CVPR.2016.90.

K. He, X. Zhang, S. Ren, and J. Sun. Identity Mappings in Deep Residual Networks. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 630–645, Cham, 2016b. Springer International Publishing. ISBN 978-3-319-46493-0. doi: 10.1007/978-3-319-46493-0_38.

W. He and Y. Chai. An Empirical Study on Energy Disaggregation via Deep Learning. In *The 2016 2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE2016)*, Beijing, China, Nov. 2016.

S. Henriet, U. Şimşekli, B. Fuentes, and G. Richard. A generative model for non-Intrusive load monitoring in commercial buildings. *Energy and Buildings*, 177: 268–278, Oct. 2018. ISSN 0378-7788. doi: 10.1016/j.enbuild.2018.07.060.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8): 1735–1780, Nov. 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

E. Holmegaard and M. B. Kjaergaard. NILM in an Industrial Setting: A Load Characterization and Algorithm Evaluation. In *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 1–8, May 2016. doi: 10.1109/SMARTCOMP.2016.7501709.

B. Humala, A. S. U. Nambi, and V. R. Prasad. UniversalNILM: A Semi-supervised Energy Disaggregation Framework Using General Appliance Models. In *Proceedings of the Ninth International Conference on Future Energy Systems*, E-Energy '18, pages 223–229, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5767-8. doi: 10.1145/3208903.3208945.

S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In F. Bach and D. Blei, editors, *Proceedings of the 32\textsuperscriptnd International Conference on Machine Learning (ICML)*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, July 2015. PMLR.

P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. Nov. 2016.

A. Iwayemi and C. Zhou. SARAA: Semi-Supervised Learning for Automated Residential Appliance Annotation. *IEEE Transactions on Smart Grid*, 8(2):779–786, Mar. 2017. ISSN 1949-3053, 1949-3061. doi: 10.1109/TSG.2015.2498642.

D. Jacobs. The German Energiewende – History, Targets, Policies and Challenges. *Renewable Energy Law and Policy Review*, 3(4):223–233, 2012. ISSN 1869-4942.

A. Jeerige, D. Bein, and A. Verma. Comparison of Deep Reinforcement Learning Approaches for Intelligent Game Playing. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0366–0371, Jan. 2019. doi: 10.1109/CCWC.2019.8666545.

K. E. H. Jenkins and D. Hopkins. *Transitions in Energy Efficiency and Demand (Open Access): The Emergence, Diffusion and Impact of Low-Carbon Innovation*. Routledge, Dec. 2018. ISBN 978-1-351-12724-0.

L. Jiang, J. Li, S. Luo, J. Jin, and S. West. Literature review of power disaggregation. In *Proceedings of 2011 International Conference on Modelling, Identification and Control*, pages 38–42, June 2011. doi: 10.1109/ICMIC.2011.5973672.

T. Joachims. Transductive Inference for Text Classification Using Support Vector Machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 200–209, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-612-8.

M. Kahl, A. U. Haq, T. Kriechbaumer, and H.-A. Jacobsen. WHITED-A Worldwide Household and Industry Transient Energy Data Set. 2016.

K. K. Kaman, M. Faramarzi, S. Ibrahim, and M. A. M. Yunus. Artificial Neural Network for Non-Intrusive Electrical Energy Monitoring System. *Indonesian Journal of Electrical Engineering and Computer Science*, 6(1):124–131, Apr. 2017.

A. Kavousian, R. Rajagopal, and M. Fischer. Ranking appliance energy efficiency in households: Utilizing smart meter data and energy efficiency frontiers to estimate and identify the determinants of appliance energy efficiency in residential buildings. *Energy and Buildings*, 99:220–230, July 2015. ISSN 0378-7788. doi: 10.1016/j.enbuild.2015.03.052.

J. Kelly and W. Knottenbelt. Neural NILM: Deep Neural Networks Applied to Energy Disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments - BuildSys '15*, pages 55–64. ACM Press, 2015a. doi: 10.1145/2821650.2821672.

J. Kelly and W. J. Knottenbelt. The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Scientific Data*, Feb. 2015b.

N. Khuriwal and N. Mishra. Breast Cancer Detection From Histopathological Images Using Deep Learning. In *2018 3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE)*, pages 1–4, Nov. 2018. doi: 10.1109/ICRAIE.2018.8710426.

H. Kim, M. Marwah, M. F. Arlitt, G. Lyon, and J. Han. Unsupervised Disaggregation of Low Frequency Power Measurements. In *Proceedings of the 11th International Conference on Data Mining*, pages 747–758, Arizona, 2010. SIAM.

D. P. Kingma and J. L. Ba. Adam: A Method for Stochastic Optimization. *ArXiv e-prints*, Dec. 2014.

J. Z. Kolter and M. J. Johnson. REDD: A public data set for energy disaggregation research. In *Proceedings of the SustKDD Workshop on Data Mining Applications in Sustainability*, San Diego, CA, USA, Apr. 2011.

J. Z. Kolter, S. Batra, and A. Y. Ng. Energy Disaggregation via Discriminative Sparse Coding. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems*, NIPS'10, pages 1153–1161, USA, 2010. Curran Associates Inc.

J. Krall, H. Street, and H. Kuhns. Graphical Closure Rules for Unsupervised Load Classification in NILM Systems. In *3rd International Workshop on Non-Intrusive Load Monitoring*, page 5, Vancouver, Canada, May 2016.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

F. P. Kuhl and C. R. Giardina. Elliptic Fourier features of a closed contour. *Computer Graphics and Image Processing*, 18(3):236–258, Mar. 1982. ISSN 0146-664X. doi: 10.1016/0146-664X(82)90034-X.

H. Y. Lam, G. Fung, and W. K. Lee. A Novel Method to Construct Taxonomy Electrical Appliances Based on Load Signaturesof. *IEEE Transactions on Consumer Electronics*, 53(2):653–660, May 2007. ISSN 0098-3063. doi: 10.1109/TCE.2007.381742.

H. Lange and M. Bergés. BOLT: Energy Disaggregation by Online Binary Matrix Factorization of Current Waveforms. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, BuildSys '16, pages 11–20, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4264-3. doi: 10.1145/2993422.2993581.

M. Längkvist, L. Karlsson, and A. Loutfi. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, 42:11–24, June 2014. ISSN 0167-8655. doi: 10.1016/j.patrec.2014.01.008.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov. 1998a. ISSN 00189219. doi: 10.1109/5.726791.

Y. LeCun, L. Bottou, G. B. Orr, and K. R. Müller. Efficient BackProp. In G. B. Orr and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade*, pages 9–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998b. ISBN 978-3-540-49430-0. doi: 10.1007/3-540-49430-8_2.

S. Leeb, S. Shaw, and J. Kirtley. Transient event detection in spectral envelope estimates for nonintrusive load monitoring. *IEEE Transactions on Power Delivery*, 10 (3):1200–1210, July 1995. ISSN 1937-4208. doi: 10.1109/61.400897.

M. Levandowsky and D. Winter. Distance between Sets. *Nature*, 234(5323):34–35, 1971. ISSN 1476-4687. doi: 10.1038/234034a0.

D. Li, K. Sawyer, and S. Dick. Disaggregating household loads via semi-supervised multi-label classification. In *2015 Annual Conference of the North American Fuzzy Information Processing Society (NAFIPS) Held Jointly with 2015 5th World Conference on Soft Computing (WConSC)*, pages 1–5, Aug. 2015. doi: 10.1109/NAFIPS-WConSC.2015.7284144.

M. Liang, Y. Meng, N. Lu, D. Lubkeman, and A. Kling. HVAC load Disaggregation using Low-resolution Smart Meter Data. In *2019 IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5, Feb. 2019. doi: 10.1109/ISGT.2019.8791578.

Z. Ling, S. Kang, H. Zen, A. Senior, M. Schuster, X. Qian, H. M. Meng, and L. Deng. Deep Learning for Acoustic Modeling in Parametric Speech Generation: A systematic review of existing techniques and future trends. *IEEE Signal Processing Magazine*, 32(3):35–52, May 2015. doi: 10.1109/MSP.2014.2359987.

B. Liu, Y. Yu, W. Luan, and B. Zeng. An unsupervised electrical appliance modeling framework for non-intrusive load monitoring. In *2017 IEEE Power Energy Society General Meeting*, pages 1–5, July 2017. doi: 10.1109/PESGM.2017.8273794.

S. Liu, M. Yamada, N. Collier, and M. Sugiyama. Change-Point Detection in Time-Series Data by Relative Density-Ratio Estimation. In G. Gimel'farb, E. Hancock, A. Imiya, A. Kuijper, M. Kudo, S. Omachi, T. Windeatt, and K. Yamada, editors, *Structural, Syntactic, and Statistical Pattern Recognition*, Lecture Notes in Computer Science, pages 363–372, Berlin, Heidelberg, 2012. Springer. ISBN 978-3-642-34166-3. doi: 10.1007/978-3-642-34166-3_40.

A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013.

S. Makonin. *Real-Time Embedded Low-Frequency Load Disaggregation*. PhD, Simon Fraser University, School of Computing Science, 2014.

S. Makonin. Investigating the switch continuity principle assumed in Non-Intrusive Load Monitoring (NILM). *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2016. doi: 10.1109/CCECE.2016.7726787.

S. Makonin and F. Popowich. Nonintrusive Load Monitoring (NILM) Performance Evaluation: A unified approach for accuracy reporting. *Energy Efficiency*, 8(4): 809–814, 2015.

S. Makonin, F. Popowich, L. Bartram, B. Gill, and I. V. Bajic. AMPds: A Public Dataset for Load Disaggregation and Eco-Feedback Research. In *Electrical Power and Energy Conference (EPEC), 2013 IEEE*, pages 1–6, Aug. 2013. doi: 10.1109/EPEC.2013.6802949.

B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2): 442–451, Oct. 1975. ISSN 0005-2795. doi: 10.1016/0005-2795(75)90109-9.

L. Mauch and B. Yang. A new approach for supervised power disaggregation by using a deep recurrent LSTM network. In *In Proceedings of the 3\textsuperscriptrd IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 63–67, Dec. 2015. doi: 10.1109/GlobalSIP.2015.7418157.

L. Mauch and B. Yang. A novel DNN-HMM-based approach for extracting single loads from aggregate power signals. In *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2384–2388, Mar. 2016.

L. Mauch, K. S. Barsim, and B. Yang. How well can HMM model load signals. In *Proceedings of the 3rd International Workshop on Non-Intrusive Load Monitoring (NILM)*, Vancouver, Canada, May 2016.

C. E. Metz. Basic principles of ROC analysis. *Seminars in Nuclear Medicine*, 8(4): 283–298, 1978. ISSN 0001-2998. doi: http://dx.doi.org/10.1016/S0001-2998(78)80014-2.

M. N. Meziane, A. Hacine-Gharbi, P. Ravier, G. Lamarque, J.-C. L. Bunetel, and Y. Raingeaud. Electrical Appliances Identification and Clustering using Novel Turn-on Transient Features:. In *Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods*, volume 2, pages 647–654, Porto, Portugal, 2017. SCITEPRESS - Science and Technology Publications. ISBN 978-989-758-222-6. doi: 10.5220/0006245706470654.

Y. Miao, M. Gowayyed, and F. Metze. EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 167–174, Dec. 2015. doi: 10.1109/ASRU.2015.7404790.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

A. S. Modi. Review Article on Deep Learning Approaches. In *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 1635–1639, June 2018. doi: 10.1109/ICCONS.2018.8663057.

V. Nair and G. E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814. Omnipress, 2010.

P. Nascimento. *Applications of Deep Learning Technologies on NILM*. PhD thesis, Alberto Luiz Coimbra Institute for Graduate Studies and Research in Engineering (COPPE), Brazil, Apr. 2016.

G. R. Newsham and C. L. Donnelly. A model of residential energy end-use in Canada: Using conditional demand analysis to suggest policy options for community energy planners. *Energy Policy*, 59:133–142, Aug. 2013. ISSN 0301-4215. doi: 10.1016/j.enpol.2013.02.030.

K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, 39(2):103–134, May 2000. ISSN 1573-0565. doi: 10.1023/A:1007692713085.

O. Parson, S. Ghosh, M. Weal, and A. Rogers. An unsupervised training method for non-intrusive appliance load monitoring. *Artificial Intelligence*, 217:1–19, Dec. 2014. ISSN 0004-3702. doi: 10.1016/j.artint.2014.07.010.

O. Parson, G. Fisher, A. Hersey, N. Batra, J. Kelly, A. Singh, W. Knottenbelt, and A. Rogers. Dataport and NILMTK: A building data set designed for non-intrusive load monitoring. In *1st International Symposium on Signal Processing Applications in Smart Buildings at 3rd IEEE Global Conference on Signal &amp; Information Processing (14/12/15 - 16/12/15)*, Sept. 2015. Num Pages: 5.

S. N. Patel, T. Robertson, J. A. Kientz, M. S. Reynolds, and G. D. Abowd. At the Flick of a Switch: Detecting and Classifying Unique Electrical Events on the Residential Power Line (Nominated for the Best Paper Award). In J. Krumm, G. D. Abowd, A. Seneviratne, and T. Strang, editors, *UbiComp 2007: Ubiquitous Computing*, Lecture Notes in Computer Science, pages 271–288, Berlin, Heidelberg, 2007. Springer. ISBN 978-3-540-74853-3. doi: 10.1007/978-3-540-74853-3_16.

M. J. D. Powell. Restart procedures for the conjugate gradient method. *Mathematical Programming*, 12(1):241–254, Dec. 1977. ISSN 1436-4646. doi: 10.1007/BF01593790.

D. M. W. Powers. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness, and correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011.

L. Prechelt. Early stopping — but when? In *Neural Networks: Tricks of the Trade: 2nd Edition*, pages 53–67. Springer, Berlin, Heidelberg, 2012. ISBN 978-3-642-35289-8.

A. Radford, L. Metz, and S. Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv:1511.06434 [cs]*, Jan. 2016.

W. M. Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. ISSN 01621459. doi: 10.2307/2284239.

A. Rasmus, T. Raiko, and H. Valpola. Denoising autoencoder with modulated lateral connections learns invariant representations of natural images. Mar. 2015a.

A. Rasmus, H. Valpola, M. Honkala, M. Berglund, and Tapani. Semi-Supervised Learning with Ladder Networks. Nov. 2015b.

REN21. Renewables in Cities 2019 Global Status Report. Technical report, REN21 Renewables Now, 2019.

O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 9351:234–241, 2015.

C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-Supervised Self-Training of Object Detection Models. In *Seventh IEEE Workshop on Applications of Computer Vision*, volume 1, pages 29–36, Jan. 2005. doi: 10.1109/ACVMOT.2005.107.

N. Roy, N. Pathak, and A. Misra. Fine-grained appliance usage and energy monitoring through mobile and power-line sensing. *Pervasive and Mobile Computing*, 30:132–150, Aug. 2016. ISSN 1574-1192. doi: 10.1016/j.pmcj.2016.01.003.

A. Ruano, A. Hernandez, J. Ureña, M. Ruano, and J. Garcia. NILM Techniques for Intelligent Home Energy Management and Ambient Assisted Living: A Review. *Energies*, 12(11):2203, June 2019. ISSN 1996-1073. doi: 10.3390/en12112203.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Representations by Back-Propagating Errors. *Nature*, 323(6088):533–536, 1986. ISSN 1476-4687. doi: 10.1038/323533a0.

A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4967–4976. Curran Associates, Inc., 2017.

J. Schleich, M. Klobasa, and S. Gölz. Does smart metering reduce residential electricity demand? In *2012 9th International Conference on the European Energy Market*, pages 1–4, May 2012. doi: 10.1109/EEM.2012.6254779.

J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, Jan. 2015. ISSN 0893-6080. doi: 10.1016/j.neunet.2014.09.003.

B. Schoelkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. Mooij. On Causal and Anticausal Learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 1255–1262, Edinburgh, Scotland, June 2012.

G. Schröder, M. Thiele, and W. Lehner. Setting Goals and Choosing Metrics for Recommender System Evaluations. In *UCERSTI2 Workshop at the 5th ACM Conference on Recommender Systems*, volume 23, Chicago, USA, 2011.

H. Scudder. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371, July 1965. ISSN 0018-9448, 1557-9654. doi: 10.1109/TIT.1965.1053799.

B. Settles. *Active Learning*. Morgan & Claypool Publishers, San Rafael, Calif., July 2012. ISBN 978-1-60845-725-0.

A. Shrestha and A. Mahmood. Review of Deep Learning Algorithms and Architectures. *IEEE Access*, 7:53040–53065, 2019. doi: 10.1109/ACCESS.2019.2912200.

R. Socher, Y. Bengio, and C. D. Manning. Deep Learning for NLP (Without Magic). In *Tutorial Abstracts of ACL 2012*, ACL '12, pages 5–5, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, Jan. 2014. ISSN 1532-4435.

R. Streubel and B. Yang. Identification of electrical appliances via analysis of power consumption. In *2012 47th International Universities Power Engineering Conference (UPEC)*, pages 1–6, Sept. 2012. doi: 10.1109/UPEC.2012.6398559.

F. Sultanem. Using appliance signatures for monitoring residential loads at meter panel level. *IEEE Transaction on Power Delivery*, 6(4):1380–1385, Oct. 1991.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, June 2015. doi: 10.1109/CVPR.2015.7298594.

G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M. A. Nicolaou, B. Schuller, and S. Zafeiriou. Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5200–5204, Mar. 2016. doi: 10.1109/ICASSP.2016.7472669.

H. Valpola. Chapter 8 - From neural PCA to deep unsupervised learning. In E. Bingham, S. Kaski, J. Laaksonen, and J. Lampinen, editors, *Advances in Independent Component Analysis and Learning Machines*, pages 143 – 171. Academic Press, 2015. ISBN 978-0-12-802806-3. doi: https://doi.org/10.1016/B978-0-12-802806-3.00008-7.

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*, 11:3371–3408, Dec. 2010. ISSN 1532-4435.

H. Wan. Deep Learning: Neural Network, Optimizing Method and Libraries Review. In *2019 International Conference on Robots Intelligent System (ICRIS)*, pages 497–500, June 2019. doi: 10.1109/ICRIS.2019.00128.

B. Wild, K. S. Barsim, and B. Yang. A New Unsupervised Event Detector for Non-Intrusive Load Monitoring. In *Proceedings of the 3rd IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Orlando, Florida, USA, Dec. 2015.

X. Xie and G. Beni. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):841–847, Aug. 1991. ISSN 1939-3539. doi: 10.1109/34.85677. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

J. Xu, H. He, and H. Man. DCPE co-training for classification. *Neurocomputing*, 86: 75–85, June 2012. ISSN 0925-2312. doi: 10.1016/j.neucom.2012.01.006.

B. Yang and Z. Zhou. Joint Modeling of Device Load and User Intention. In *2011 46th International Universities' Power Engineering Conference (UPEC)*, pages 1–6, Sept. 2011.

D. Yarowsky. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA, June 1995a. Association for Computational Linguistics. doi: 10.3115/981658.981684.

D. Yarowsky. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, ACL '95, pages 189–196, Cambridge, Massachusetts, USA, 1995b. Association for Computational Linguistics. doi: 10.3115/981658.981684.

K. Yeo and I. Melnyk. Deep learning algorithm for data-driven simulation of noisy dynamical system. *Journal of Computational Physics*, 376:1212–1231, Jan. 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2018.10.024.

W. J. Youden. Index for Rating Diagnostic Tests. *Cancer*, Vol. 3(1):32–35, 1950. ISSN 1097-0142. doi: 10.1002/1097-0142(1950)3:1<32::AID-CNCR2820030106>3.0.CO;2-3.

F. Yu and V. Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. In *In Proceedings of the 4th International Conference on Learning Representations (ICLR)*, Puerto Rico, US, May 2016.

M. Zeifman and K. Roth. Viterbi algorithm with sparse transitions (VAST) for non-intrusive load monitoring. In *2011 IEEE Symposium on Computational Intelligence Applications In Smart Grid (CIASG)*, pages 1–8, Apr. 2011.

C. Zhang, M. Zhong, Z. Wang, N. Goddard, and C. Sutton. Sequence-to-Point Learning With Neural Networks for Non-Intrusive Load Monitoring. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, New Orleans, Louisiana, USA, Feb. 2018.

L. Zhang, L. Zhang, and B. Du. Deep Learning for Remote Sensing Data: A Technical Tutorial on the State of the Art. *IEEE Geoscience and Remote Sensing Magazine*, 4(2): 22–40, June 2016. doi: 10.1109/MGRS.2016.2540798.

S. Zhang, S. M. H. Bamakan, Q. Qu, and S. Li. Learning for Personalized Medicine: A Comprehensive Review From a Deep Learning Perspective. *IEEE Reviews in Biomedical Engineering*, 12:194–208, 2019. doi: 10.1109/RBME.2018.2864254.

X. Zhu and A. B. Goldberg. Introduction to Semi-Supervised Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 3(1):1–130, Jan. 2009. ISSN 1939-4608. doi: 10.2200/S00196ED1V01Y200906AIM006.

A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar. Non-Intrusive Load Monitoring Approaches for Disaggregated Energy Sensing: A Survey. *Sensors (Basel, Switzerland)*, 12(12):16838–16866, Dec. 2012. ISSN 1424-8220. doi: 10.3390/s121216838.

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Alle sinngemäß und wörtlich übernommenen Textstellen aus der Literatur bzw. dem Internet wurden unter Angabe der Quelle kenntlich gemacht.

Stuttgart, den 31.05.2021

Karim Said Barsim