



HAL
open science

Towards user-centric optimisation and predictive control approaches of the performance of smart buildings

Chuhao Jiang

► **To cite this version:**

Chuhao Jiang. Towards user-centric optimisation and predictive control approaches of the performance of smart buildings. Other. Université d'Angers, 2023. English. NNT : 2023ANGE0047 . tel-04498445

HAL Id: tel-04498445

<https://theses.hal.science/tel-04498445>

Submitted on 11 Mar 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ D'ANGERS

ÉCOLE DOCTORALE N° 602

Sciences pour l'Ingénieur

Spécialité : « *Mécanique des fluides, énergétique, thermique, combustion, acoustique* »

Par

Chuhao Jiang

«Towards user-centric optimisation and predictive control approaches of the performance of smart buildings»

Thèse présentée et soutenue à « Angers », le « 4 Décembre 2023 »

Unité de recherche : « Laboratoire Angevin de Recherche en Ingénierie des Systèmes »

Rapporteurs avant soutenance :

Didier DEFER Professor at University of Artois
Marie DUQUESNE Professor at La Rochelle University

Composition du Jury :

Président :	Monika WOLOSZYN	Professor at Savoie Mont-Blanc University
Examineurs :	Alain GODON	Associate professor at the University of Angers
	Abderafi CHARKI	Associate professor at the University of Angers
	Tingting Vogt WU	Associate professor at the University of Bordeaux
Dir. de thèse :	David BIGAUD	Professor at the University of Angers
Co-dir. de thèse :	Marie-Lise PANNIER	Associate professor at the University of Angers

ACKNOWLEDGEMENT

As I write this, a mix of emotions engulfs me. There is so much I want to express, yet words momentarily elude me. It is thanks to the love, support, and inspiration of many that I have been able to present this article to you, and I would like to extend my deepest gratitude to all of them.

First and foremost, I sincerely wish to thank my professors, David Bigaud and Marie-lise Pannier. I feel so blessed to have the best mentor in the world. Thank you both for your unwavering support, for every opportunity you've provided, and for your invaluable guidance. I'm deeply grateful for the patience you've shown and the meticulous attention you've given to my work, as well as the personal care and concern you've extended towards my life.

David, I thank you for guiding me through my academic journey. Many a time, I entered supervisory meetings filled with doubt and dejection. However, I always left with renewed confidence and a clear path forward.

Marie, I am indebted to you for patiently teaching me academic skills from scratch, and guided me with utmost care and attention. I have had so many good memories with you in every conference, and in every meeting, you were there to help me prepare in every detail. I am also deeply grateful for your help in easing my transition to life in France.

A heartfelt thank you to both of you. Your dedication, seriousness, and passion towards work and academia will continue to inspire me, setting a benchmark for my future endeavors.

I would also like to extend my gratitude to professor Alain Godon for his invaluable contributions to the design, fabrication, and guidance related to the sensor board. Thanks to him, we were able to gather a wealth of data. Additionally, I appreciate professor Monika Woloszyn and professor stephane ploix for the constructive feedback provided during every CSI meeting.

To all members of the LARIS, thank you for your help and support. Salim, I enjoy every discussions time with you, and appreciate every help from you. Nicolas, thank you for the support in both the lab and the gym; I hope to visit you in Napoli next time. Therese, thank you for guiding me during my initial times in France. And to my dear

friend Yuchao Dong, you've always been a true friend and mentor.

I'd like to express my deepest gratitude to my fiancée, Mengrui Liu, an exceptional pianist. Thank you for accompanying me throughout my doctoral journey. You are the light in my life, illuminating even the darkest moments. I appreciate your constant companionship and unwavering support. Even though I still owe you a wedding, I am ready to spend the rest of my life with you.

I want to also thank my parents who always have faith on me. I am sorry that I couldn't be by your side for the past ten years. Your unconditional and steadfast love and support mean everything to me. Without your love and support, I wouldn't never go this far. Thanks, mom and dad. Love you all.

Lastly, I'd like to say to myself:

May you brave a thousand waves, yet never forget your youthful days.

TABLE OF CONTENTS

List of Acronyms	xxiii
Introduction	1
1 Literature review	11
1.1 Performance monitoring of buildings	12
1.1.1 Energy performance indicators	13
1.1.2 Monitoring and sensors	15
1.1.3 The Optimal Sensor Placement (OSP) issue	22
1.1.4 Other Challenges in smart building	27
1.1.5 Intermediate conclusion	29
1.2 Occupants' activities, occupancy monitoring and event detection	30
1.2.1 The Occupants - Human in the Loop (HiL)	30
1.2.2 The Activities – Occupants Behavior (OB)	32
1.2.3 The monitoring of occupancy and activities	37
1.3 Detection, identification and estimation	41
1.3.1 Main categories of learning methods	42
1.3.2 Detection and identification of activities – a classification problem	43
1.3.3 Estimation and prediction – a regression problem	55
1.3.4 Some problems in using machine learning for activities and events detection, estimation/prediction	59
1.4 Performance prediction and control of buildings	62
1.4.1 Control strategy in buildings	62
1.4.2 Importance of occupancy centric control	63
1.5 Conclusion of this chapter	65
2 Presentation of the case studies	67
2.1 Introduction of the chapter	67
2.2 Simulation-based case study	68
2.2.1 Aim of the case study	68

TABLE OF CONTENTS

2.2.2	Modelling tools	69
2.2.3	Building description	70
2.2.4	Building energy modelling	71
2.2.5	Building CFD modelling	74
2.3	Real data based case study	77
2.3.1	Aim of the case study	77
2.3.2	Presentation of the monitored building	78
2.3.3	Description of the monitoring	80
2.3.4	Data recorded	88
2.4	Conclusion of the chapter	93
3	Optimal sensor location	95
3.1	Introduction	95
3.2	Complementary state of the art and theoretical aspects	96
3.2.1	The effective independent method (EIM)	97
3.2.2	The information entropy (IE) method	100
3.3	Necessary adaptations before application on simulation data with EIM and IE method	102
3.3.1	Adaptation and application of EIM	103
3.3.2	Adaptation and application of IE method	107
3.3.3	Further discussions about Optimal Sensor Placement	108
3.4	First applications and results on a single room case study	113
3.4.1	Stability analysis	116
3.4.2	Accuracy analysis	119
3.5	Result for multi-wall and multi-zone	122
3.6	Result for real case study	124
3.6.1	Essential key elements of the real case study	124
3.6.2	Finding the optimal sensor location	128
3.6.3	Prediction of physical parameters at target locations using values at optimal sensor locations	132
3.7	Conclusion of the chapter	138
4	Single event detection	139
4.1	Introduction	139
4.2	Methodologies detailed	141

4.2.1	Unsupervised DBSCAN	141
4.2.2	Unsupervised Principal Component Analysis (PCA)	146
4.2.3	Unsupervised Multivariate Gaussian distribution (MGD)	148
4.2.4	Supervised Logistic regression	149
4.2.5	Supervised Artificial neural network	150
4.2.6	Supervised XGBoost	151
4.2.7	Semi-supervised AutoEncoder	153
4.3	Case study simulation data	154
4.3.1	First case study – Imbalanced dataset	155
4.3.2	Second case study – A more realistic and balanced dataset	180
4.3.3	Occupant presence detection	185
4.3.4	An intermediate conclusion following case studies with simulation data	187
4.4	Real case study results	190
4.4.1	Presentation of the real case dataset	190
4.4.2	Comparison of learning algorithms - Main results for real case study	191
4.5	Algorithms acceleration	195
4.5.1	DBSCAN acceleration	195
4.5.2	XGBoost acceleration	198
4.6	Conclusion of the chapter	202
5	Multi activities detection based on root cause analysis	205
5.1	Introduction	205
5.2	Case study	207
5.2.1	Simulation case study	207
5.2.2	Real case study	210
5.3	Self-learning supervised method	210
5.3.1	Background of self-learning supervised method	210
5.3.2	Methodology and result for the Multi-Occupant Behavior Self-Supervision Learning	213
5.4	Combination of DBSCAN, BN and Hotspot algorithm for multiactivities detection	219
5.4.1	BN in the perspective of root cause analysis	220
5.4.2	Comparison of Root Cause Analysis algorithms and choice	230

TABLE OF CONTENTS

5.4.3	Development of a novel hybrid for occupant behavior analysis . . .	233
5.5	Conclusion of the chapter	255
Conclusions and Perspectives		257
Bibliography		263
Appendices		297
A Annex on Chapter 1		297
A.1	Energy performance indicators	297
A.2	Pros and Cons of different technologies of IoT technologies	299
A.3	Optimal sensor location algorithms	304
A.4	Taxonomies for occupant behavior	306
A.5	Strengths and weakness of occupancy detection sensors	310
A.6	Application of artificial intelligence to detect occupancy related activities and events	314
A.7	Strengths and weakness of artificial intelligence algorithms	318
A.8	Current problems with the application of machine learning algorithms for occupancy activities and event detection	326
B Annex on Chapter 5		331
B.1	Review of self-learning supervised method	331
B.2	Estimation BN architecture	333
B.2.1	Score based structure estimation	334
B.2.2	Constraint-based structure estimation	337
B.2.3	Hybrid structure estimation	338
B.2.4	Domain knowledge	340
B.3	CPD parameters learning methods	340
B.3.1	Maximum Likelihood Estimation (MLE)	340
B.3.2	Bayesian Estimation	341
B.4	Application of the BN to the real case study	343
B.4.1	Bayesian network architecture for real case	343
B.4.2	Joint Probability Distribution in BNs in real case	344
B.4.3	Variable elimination real case	346
B.5	RCA heuristic search algorithms	347

B.5.1	Adtributor	347
B.5.2	Hotspot	350
B.6	Real case BN result	353
C	Annex on Perspectives	357
C.1	Methodology proposal for model predictive control in this study	357
C.2	Building model	358
C.2.1	Grey box model	361
C.2.2	Parameter estimation	362
C.2.3	Model validation	364
C.3	User information integration	365

Nomenclature

List of Acronyms

Acronym	Explanation
AAEC	Activity Appliance Energy Consumption
AdaGrad	Adaptive Gradient Algorithm
Adam	Adaptative Moment Estimation
AE	Auto-Encoders
AF	Adaptive Filtering
AHU	Air Handling Unit
AI	Artificial Intelligence
ALM	Appliance load monitoring
ALRCN	Auto-encoder Long-term Recurrent Convolutional Network
AMI	Advanced Metering Infrastructure
ANN	Artificial Neural Network
ARIMA	Auto-Regressive Integrated Moving Average
ARMA	Auto-Regressive and Moving Average
ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Engineers
AUC	Area Under the Curve
BAS	Building Automation Systems
BEI	Building Energy Index
BBP	Batch Back Propagation
BEMS	Building Energy Management System
BIC	Bayesian Information Criterion
BIoT	Building Internet of Things
BLE	Bluetooth Low Energy
BMS	Building Management system
BN	Bayesian Network
BP	Back Propagation
BTU	British Thermal Units
C	Outside Temperature
c	CO ₂ level
CAF	Cross Ambiguity Function
CART	Classification and Regression Tree

Acronym	Explanation
CFD	Computational Fluid Dynamics
CNN	Convolutional Neural Networks
CPD	Conditional Probability Distribution
CPT	Conditonal Probability Tables
CSA	Connectivity Standatds Alliance
DAG	Directed Acyclic Graph
DBES	Dynamic Building Energy Simulations
DBN	Deep Belief Networks
DBN	Dynamic Bayesian Networks
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DL	Deep Learning
DNAs	Drivers, Needs, Actions, and Systems
DOF	Degree of Freedom
DR	Demand Response
E	Electricity
EBT	Ensemble Bagging Tree
ECG	Electrocardiogram
ECI	Energy Conservative Index
ECI	Energy Cost Index
ED	Effective Independence Coefficient
EDI	Energy Demand Intensity
EEI	Energy Efficiency Index
EI	Effective Independence
EIM	Effective Independence Method
EL	Ensemble Learning
ELM	Extreme Learning Machine
EM	Expectation-Maximization
EMC	Energy Management Controller
EMS	Energy Management System
EnPI	Energy Performance Indicator
ESS	Equivalent Sample Size
EUI	Energy Use Index
FDD	Fault Detection and Diagnosis

TABLE OF CONTENTS

Acronym	Explanation
FEM	Finite Element Model
FIM	Fisher Information Matrix
FP	False positive
FPR	False Positive Rate
FN	False negative
GA	Genetic Algorithm
GAN	Generative Adversarial Network
GBM	Gradient Boosting Machine
GMM	Gaussian Mixture Model
GMP	Gaussian Mixture Process
H	Heating (simulation case)/Humidity (real case)
HEMS	Home Energy Management Systems
HERS	Home Energy Rating System
HiL	Human integration in the Loop
HMM	Hidden Markov Model
HVAC	Heating, Ventilation, and Air Conditioning
I	Indoor temperature (simulation case)/Light Intensity (real case)
IEA	International Energy Agency
IECC	International Energy Conservation Code
IESR	Instant Energy Scheduling Recommendation
IoT	Internet of Things
IE	Information Entropy
kL	Kullback-Leibler
k-NN	k-Nearest Neighbors
KPI	Key Performance Indicator
LAN	Local Area Network
LC	Learning curve
LDA	Linear Discriminant Analysis
LDEM	Local Density Estimation
LI	Linear Interpolation
LM	Levenberg-Marquardt
LSM	Least Mean Squares
LSTM	Long Short-Term Memory

Acronym	Explanation
LR	Logistic Regression
MB	Markov Blanket
MCAR	Missing Completely at Random
MCTS	Monte Carlo Tree Search
M-FRNN	Markov-based Feedback Recurrent Neural Networks
MGD	Multivariate Gaussian Distribution
MI	Myocardial Infarction
MIMO	Multiple-Input-Multiple-Output
ML	Machine Learning
MLP	Multilayer Perceptron
MMHC	Hybrid structure estimation / Max-Min Hill-Climbing
MMPC	Max-Min Parents and Children
MPC	Model Predictive Control
MSE	Mean Squared Error
MST	Maximum Spanning Tree
NB	Naïve Bayes
NBC	Naïve Bayes Classification
NLP	Natural Language Processing
O	Occupancy(simulation case)/TVOC level(real case)
OB	Occupants Behavior
O _i	Outdoor Weather
O&M	Operations and Maintenance
OPTICS	Ordering Points to Identify The Clustering Structure
OSP	Optimal Sensors Placement
O _u	Outdoor Temperature(simulation case)/Sound Level (real case)
PC	Constraint-based structure estimation / Parents and Childrens
PCA	Principal Component Analysis
PED	Peak Energy Demand
PI-PRM	Physics-Informed Pattern-Recognition Machine
PIR	Pyroelectric InfraRed
PLC	PowerLine Communication
PSO	Particle Swarm Optimization
PTZ	Pan-Tilt-Zoom

TABLE OF CONTENTS

Acronym	Explanation
PWR	Passive Wi-Fi Radar
QI	Quadratic Interpolation
QP	Quick Propagation
Q-Q	Quantile-Quantile
RBC	Rule-Based Control
RC	Resistor-Capacitor
RCA	Root Cause Analysis
ReLu	Rectified Linear Unit
RF	Radio Frequency
RFo	Random Forests
RFID	Radio Frequency Identification
RL	Reinforcement Learning
RNN	Recurrent Neural Networks
ROC	Receiver Operating Characteristic
RMSE	Root Mean Square Error
RMSProp	Root Mean Square Propagation
RSSI	Received Signal Strength Indicator
S	Solar(simulation case)/Temperature (real case)
SARIMA	Seasonal AutoRegressive Integrated Moving Average
SC	Spectral Clustering
SDTL	Seasonal Decomposition of Time Series
SGD	Stochastic Gradient Descent
SHM	Structural Health Monitoring
SMOTE	Synthetic Minority Oversampling TEchnique
SNN	Shallow Neural Networks
SPE	Squared Prediction Error
SVM	Support Vector Machines
SVR	Support Vector Regression
T	Temperature
TABS	Thermally Activated Building System
ToF	Time-of-Flight
TN	True Negative
TPR	True Positive Rate

Acronym	Explanation
t-SNE	t-Distributed Stochastic Neighbor Embedding
TSVM	Transductive Support Vector Machines
UMAP	Uniform Manifold Approximation and Projection
VAE	Variational Auto-encoder
VAV	Variable Air Volume
VE	Variable Elimination
VOC	Volatile Organic Compound
W	Window Open
WSN	Wireless sensor network
XGBoost	eXtreme Gradient Boosting
XLM	EXtreme Learning Machine
zEPI	zero Energy Performance Index

LIST OF FIGURES

1	Global energy share of buildings and construction	2
1.1	Description of Advanced Metering Infrastructure (AMI)	21
1.2	Sensor faults and classification (Bae et al., 2021)	28
1.3	Illustration of under and over -sampling techniques for imbalanced data.	61
1.4	Illustration of Synthetic Minority Over-sampling Technique – SMOTE - (Chawla et al., 2002)	62
2.1	Fictitious building modelling in DesignBuilder.	70
2.2	A week occupancy schedule from the generated scenario.	72
2.3	Evolutions of the indoor and outdoor temperatures during the simulated year.	73
2.4	Grid resolution.	75
2.5	CFD simulation results for January 1st at 12 am.	77
2.6	Site plan of Polytech Angers. Adapted from Google Maps.	78
2.7	Picture and sketch of room 114.	79
2.8	Picture and sketch of room 219.	79
2.9	Multiphysical sensor boards’ assembly workshop and completed board.	81
2.10	Multiphysical sensor locations in room 114 (a) and 219 (b).	83
2.11	Multiphysical sensor board at student’s locations in the classroom 219.	84
2.12	Weather station installed on the roof (a) and data extraction module (b).	84
2.13	A sensor placed on a door (upper right corner).	86
2.14	Names and locations of windows and doors sensors in room 114 (a) and 219 (b).	86
2.15	Magnetic core (a) and recording device (b) to monitor the electricity con- sumption.	87
2.16	Survey and information on the monitoring available in the rooms.	89
2.17	Comparison of the RMSE for some CO ₂ sensors of room 219 using the MCAR missing data generation mechanism and for 10 % (a) and 60 % (b) of missing data. From (Es-Sabar 2022).	90

2.18	Comparison of the RMSE on the final regression task using the MCAR missing data generation mechanism for 10 % (a) and 60 % (b) of missing data. Adapted from (Es-Sabar 2022).	91
2.19	Number of occupants	92
2.20	CO ₂ concentration	92
2.21	Indoor temperature	92
2.22	Humidity	92
2.23	Total VOC	92
2.24	Light	92
2.25	Electricity consumption	92
2.26	Opened windows	92
2.27	Outdoor temperature	92
2.28	Sun radiation	92
3.1	Layout of the room and sensors (m = 528 candidate), targets (n = 9) locations	102
3.2	Comparison of optimal sensor locations found from EIM and IE methods	114
3.3	Comparison of multiple optimal sensor locations from EIM and IE methods	115
3.4	Stability evaluation of the EIM	117
3.5	Stability evaluation of IE method	118
3.6	information entropy for original and noisy data	119
3.7	RMSE of the value optimal sensor in EIM with target value	120
3.8	RMSE of the value optimal sensor in IE methods with target value	121
3.9	At the left, a larger image of the point on Hot wall	122
3.10	At the right, a larger image of the point on Cold wall	122
3.11	Result for multi-wall configuration	122
3.12	At the left, a larger image of the point on zone left	123
3.13	At the right, a larger image of the point on zone right	123
3.14	Result for multi-zone	123
3.15	Layout of the sensors in Real case	125
3.16	CO ₂ variation measured at different sensors and targets	127
3.17	Humidity variation measured at different sensors and targets	128
3.18	Temperature variation measured at different sensors and targets	128
3.19	Top three optimal sensor location for each method	130
3.20	RMSE for all sensors with regard to target point 1097BD2A4A8	134
3.21	RMSE for all sensors with regard to target point 1097BD2E2CC	135

LIST OF FIGURES

3.22 Sensor 100 CO₂ VS target CO₂ 137

3.23 Sensor 100 Temperature VS target Temperature 137

3.24 Sensor 101 CO₂ VS target CO₂ 137

3.25 Sensor 101 Temperature VS target Temperature 137

3.26 Sensor 102 CO₂ VS target CO₂ 137

3.27 Sensor 102 Temperature VS target Temperature 137

3.28 Sensor 103 CO₂ VS target CO₂ 137

3.29 Sensor 103 Temperature VS target Temperature 137

3.30 Sensor 104 CO₂ VS target CO₂ 137

3.31 Sensor 104 Temperature VS target Temperature 137

3.32 Sensor 105 CO₂ VS target CO₂ 137

3.33 Sensor 105 Temperature VS target Temperature 137

3.34 Sensor 106 CO₂ VS target CO₂ 137

3.35 Sensor 106 Temperature VS target Temperature 137

3.36 Sensor 107 CO₂ VS target CO₂ 137

3.37 Sensor 107 Temperature VS target Temperature 137

3.38 Sensor 111 CO₂ VS target CO₂ 137

3.39 Sensor 111 Temperature VS target Temperature 137

3.40 Sensor 112 CO₂ VS target CO₂ 137

3.41 Sensor 112 Temperature VS target Temperature 137

4.1 Illustration of core, border and noise points associated with DBSCAN algorithm (from Amini et al. (2014)). 143

4.2 Anomaly detection process of AutoEncoder [from Youngrok et al. (2021)] . 154

4.3 UMAP of simulation dataset: green (closed/normal) and red (open/abnormal) dots show data imbalance 156

4.4 Grid search for DBSCAN parameters chosen 160

4.5 Grid search for PCA and MGD threshold chosen 161

4.6 ROC curve for three algorithms 163

4.7 K-distance graph 166

4.8 PCA reconstruction error VS instance 167

4.9 Histogram of MGD scores 168

4.10 ANN structure 171

4.11 LC logistic regression 172

4.12 LC XGBoost 172

4.13	LC ANN	172
4.14	Logistic regression Undersampling	174
4.15	Logistic regression Oversampling	174
4.16	Logistic regression SMOTE	174
4.17	5 folder cross validation	174
4.18	Violin plot	175
4.19	AutoEncoder architecture	177
4.20	Finding the best threshold for AutoEncoder	179
4.21	UMAP window status for balanced case	181
4.22	Learning curve ANN	184
4.23	UMAP in real data	191
4.24	Features distribution	193
4.25	Features distribution	193
4.26	Features distribution	193
4.27	Features distribution	193
4.28	Architecture of XGBoost-Ray	200
4.29	XGBoost-Ray training actors	201
5.1	Electricity simulation case	209
5.2	Heat consumption simulation case	209
5.3	Indoor temperature simulation case	209
5.4	Occupancy status simulation case	209
5.5	Outdoor temperature simulation case	209
5.6	Solar gain simulation case	209
5.7	Window status simulation case	209
5.8	Window door sensor and selected sensor	210
5.9	Supervised VS unsupervised VS self supervised learning (Liu et al.2023)	211
5.10	Self supervision learning steps	216
5.11	Three algorithms with low percentage labeled data	218
5.12	Self supervision learning confusion matrix	218
5.13	BN structure comparison for simulated case study	223
5.14	BN structure of the simulation case study	224
5.15	Flow chart for multi activities detection	235
5.16	ANN for simulation	236
5.17	ANN prediction for the simulation case	237

LIST OF FIGURES

5.18	Q-Q plot heat consumption	240
5.19	Q-Q plot indoor temperature	240
5.20	Q-Q plot Occupancy	240
5.21	Q-Q plot Outdoor temperature	240
5.22	Q-Q plot Electricity	240
5.23	Q-Q plot Window open	240
5.24	Q-Q plot solar gain	240
5.25	4-D cuboids	243
5.26	BN for occupancy status inference	248
5.27	Occupancy status real VS inference	248
5.28	BN for window status inference	249
5.29	Window status real VS inference	249
5.30	Extracted one day BN inference	250
5.31	Detected faults	251
5.32	Inference probability VS ground truth	252
5.33	Zoom in inference probability	253
5.34	Confusion matrix for the ANN-DBSCAN-HOTSPOT-BN	254
B.1	Figure rotation pretext	332
B.2	prior distribution of a variable	344
B.3	BN structure of the real case study	345
B.4	BN for real case window status inference	354
B.5	Real case window status VS inference	354
C.1	MPC example	358
C.2	RC model 5R3C	361
C.3	Parameter estimation flow chart	363

LIST OF TABLES

1.1	Confusion Matrix	44
2.1	External walls composition	70
2.2	Roof composition	71
2.3	Floor composition	71
2.4	Measurement characteristics of each ambient sensor	82
2.5	Measurement characteristics of the weather station, from the provided documentation	85
2.6	Measurement deployment in rooms 114 and 219	87
3.1	Missing data for each parameter and sensor	126
3.2	Condition numbers and priority for EIM	131
3.3	Parameter entropy and total entropy	132
3.4	Combined RMSE for each wall sensor to target point 1097BD2A4A8	134
3.5	Combined RMSE for each wall sensor to target point 1097BD2E2CC	135
4.1	Machine Learning Methods	140
4.2	Comparison of Precision, recall, and F1 Score for unsupervised DBSCAN, PCA, and MGD methods.	158
4.3	Comparison of confusion matrix results for DBSCAN, PCA, and MGD methods	159
4.4	Comparison of Original and New Thresholds and Parameters for DBSCAN, PCA, and MGD	162
4.5	Comparison of precision, recall, and F1 Score for DBSCAN, PCA, and MGD methods after Grid search	162
4.6	Comparison of confusion matrix results and computation times for DBSCAN, PCA, and MGD methods after Grid search	162
4.7	Performance Metrics for Different Models and Sampling Techniques	176
4.8	Balanced Accuracy and ROC AUC Score for Different Models and Sampling Techniques	176

LIST OF TABLES

4.9 Confusion matrix for AutoEncoder 178

4.10 Performance Metrics for AutoEncoder 178

4.11 Confusion matrix for AutoEncoder with optimal threshold 179

4.12 Performance Metrics for AutoEncoder with optimal threshold 179

4.13 Comparison of confusion-matrix-based indicators for DBSCAN, PCA, and
MGD methods after Grid search 182

4.14 Comparison of different algorithm precision, Recall and F1 182

4.15 Comparison of Performance Metrics for Different Models and Sampling
Techniques 183

4.16 Confusion matrix for AutoEncoder in balanced simulation data 184

4.17 Performance Metrics for AutoEncoder in balanced simulation data 185

4.18 Comparison of confusion matrix results for DBSCAN, PCA, and MGD
methods after Grid search 185

4.19 Comparison of different algorithm precision, Recall and F1 185

4.20 Comparison of Performance Metrics for Different Models and Sampling
Techniques 187

4.21 Confusion matrix for the AutoEncoder for presence detection 187

4.22 Performance Metrics for the AutoEncoder for presence detection 187

4.23 Comparison of confusion matrix results for DBSCAN, PCA, and MGD
methods after Grid search 191

4.24 Comparison of different algorithm precision, recall and F1 in real data . . . 192

4.25 Comparison of Performance Metrics for Different Models and Sampling
Techniques 194

4.26 Confusion matrix for the AutoEncoder for real case 194

4.27 Performance metrics for the AutoEncoder for real case 194

4.28 Comparison of Standard and Optimized Search on Simulation Dataset . . . 198

4.29 Performance metrics for different training methods across various datasets. 201

5.1 Algorithm Accuracy Comparison 217

5.2 Variables for the simulation case study 221

5.3 Comparison of accuracies between Maximum Likelihood Estimation and
Bayesian Estimation in simulation case study 226

5.4 Summary of heuristic search methods 232

5.5 Simulation window open confusion matrix 237

5.6 Simulation presence of occupancy confusion matrix 238

5.7	Attributes and Heat value	242
5.8	Attributes and real/predicted Heat (Discretized)	242
5.9	Discrepancies between actual and predicted KPI	245
5.10	Root Causes	252
A.1	Description, strengths, and weaknesses of monitoring and data collecting and transferring technologies.	300
A.2	Summary of optimal sensor location algorithms	305
A.3	Example of taxonomy for occupant behavior	307
A.4	Other example of taxonomy for occupant behavior	309
A.5	Strengths and weaknesses of some sensors used for building occupancy estimation.	311
A.6	Examples of applications, with information on the different types of sensor data used, the methods used and their accuracy.	315
A.7	Examples of models, with their main features, categories, type of data, complexity, strengths and weaknesses.	320
A.8	Some problems associated with the use of machine learning for classification and regression and examples of methods most and least sensitive to the identified weaknesses.	327
B.1	Definitions and Notations	351
B.2	Merged Inferred Probabilities and Ground Truth	355

INTRODUCTION

Energy has always been a critical factor in the history of economic and social growth, serving as an essential component of human life (Lloyd, 2017). However, the world is currently facing a daunting energy dilemma as the demand for energy continues to rise rapidly, while the supply of energy is becoming increasingly constrained due to environmental and economic factors. Achieving a balance between demand and supply while reducing environmental impact has become a global concern (Gan et al.2020).

According to the International Energy Agency (IEA), in 2020, the world consumed approximately 170,000 TeraWatt-hours (TWh) of energy, with fossil fuels accounting for about two-thirds of that total (IEA, 2020) ("Global Energy Review 2020", 2020). This trend is expected to continue in the coming decades, with the IEA predicting a more than 30% increase in global energy consumption between 2020 and 2040, largely driven by growing countries (IEA, 2020) ("Global Energy Review 2020", 2020). This projected increase in energy consumption is expected to put a strain on the current energy systems and resources, necessitating significant investments in new technologies, energy infrastructure, and energy efficiency.

The sources of energy, including fossil fuels, nuclear energy, hydropower, and renewable energy, have a significant impact on energy infrastructure and the environment, particularly on climate change ("Energy and Climate Change — European Environment Agency," EEA Signals 2017). Climate change refers to long-term changes in the Earth's average temperature and weather patterns caused by the increase in greenhouse gas emissions resulting from society's development and use of fossil fuels. Greenhouse gases, such as carbon dioxide (CO₂), trap heat in the atmosphere, leading to global temperature rise, melting of glaciers and ice caps, thermal expansion of seawater, rising sea levels, increased frequency and intensity of extreme weather events, and other detrimental effects on infrastructure, agriculture, and human health. Therefore, it is imperative to change our energy consumption and production practices to reduce their impact on the environment and limit global temperature increase to below 2°C above pre-industrial levels, as advocated by the Intergovernmental Panel on Climate Change (IPCC) (IPCC, 2023) (IPCC, 2022).

To achieve a balance between demand and supply while mitigating environmental im-

pact, it is crucial to transition to sustainable energy sources and improve energy efficiency. To tackle these issues, improving the energy efficiency of buildings has emerged as a critical area of focus. As depicted in Figure 1, buildings account for a significant portion of global energy consumption and greenhouse gas emissions, making them a key target for improving energy performance.

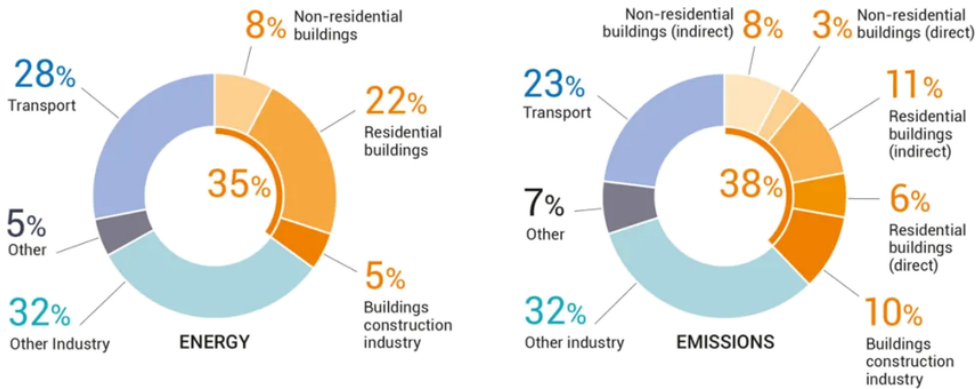


Figure 1 – Global energy share of Buildings and construction (From IEA 2021a)

There are several levers that can be used to reduce the energy consumption of buildings. The easiest to implement refer to sobriety, which includes the involvement of occupants in energy management. Then energy efficiency measure can be taken, such as, the use of high-performance materials, the design of optimized products and systems, the measurement, monitoring and analysis of performance indicators, the regulation of energy systems through intelligent technologies. Finally, after having significantly reduced the consumption through sobriety and measures to improve efficiency, renewable energy sources can be integrated to cover the residual demand

Prior to the deployment of renewables, effective approaches to improve the energy efficiency of buildings are the use of high-performance materials, products, and systems. Insulation is a key factor that can significantly enhance energy efficiency in buildings. High-performance insulation materials, such as spray foam insulation, can greatly reduce heat loss and help maintain a constant temperature inside the building. This means that it can effectively reduce heat transfer and energy consumption for heating and cooling purposes. Additionally, building thermal mass can serve as a good energy storage system (Y. Chen et al. 2020). It can provide an energy storage solution to address the intermittency problem of renewable energy, and it also can balance the demand and production sides. For example, Turner et al. (2015) studied mechanical pre-cooling for load shifting in residential buildings, and the results showed that at least 50% of the on-peak residential

electricity consumption from mechanical cooling can be shifted to the off-peak period by utilizing the building's thermal mass. Windows and doors are also important products that can enhance the energy efficiency of buildings. Energy-efficient windows and doors are designed to prevent heat loss and gain, thereby reducing the amount of energy needed for heating and cooling. Examples of energy-efficient features that can be incorporated into windows and doors include low-emissivity coatings, gas fills, and insulated frames. Heating, ventilation, and air conditioning (HVAC) systems are major energy consumers in buildings, and using high-performance HVAC systems can greatly reduce energy consumption and operating costs. Energy-efficient HVAC systems are designed to deliver heating and cooling more efficiently, improving indoor air quality and providing better control over temperature and humidity levels. Techniques such as regenerative evaporative cooling and heat recovery can also be used to reduce energy consumption in HVAC systems. For example, regenerative evaporative cooling technology can reduce the load on the chiller system and result in lower power usage, leading to energy savings. Heat recovery techniques, such as using enthalpy/membrane heat exchangers or energy recovery ventilators, can transfer heat between fluids and reduce annual energy costs.

With the development of the smart building, Building Energy Management Systems (BEMS) have a crucial role to play in shaping energy consumption in buildings, adjusting it to align more closely with the need of building occupants. Numerous studies have confirmed the substantial impact of occupant's presence and behavior on the energy efficiency of buildings (Bazazzadeh et al. 2021, Paone et al.2018, Uddin et al.2021). Factors such as occupant's behavior, the efficiency of control services, and any deviations from the design quality of the building all contribute significantly to the waste of energy during the operational phase of a building. Research underscored that energy consumption could swing drastically based on the occupancy.

The recent literature underlines the urgent need to develop and implement more occupant-focused control systems. These occupant-centric systems could potentially lead to significant energy savings by adjusting the operation of building systems like HVAC, lighting, and others in response to the real-time presence and behavior of occupants. Ultimately, the efficient operation of BEMS depends on the complex interplay between building design, control systems, and human behavior. Thus, future research and development in this field should focus on integrating these aspects to optimize energy use in buildings. Presently, the optimization of energy usage in buildings primarily concentrates on the influences of the external environment and built-in equipment, with less consideration

given to occupant's behavior. In certain cases, standardized profiles of occupant behavior are used as proxies for real-world behavior rather than investigating and accounting for actual occupant behaviors and the myriad uncertainties associated with them. Needless to say, using such an approach often results in a gap between the predicted and actual energy consumption. Existing building systems, for instance, the HVAC system, typically operate on a survey or expert determined schedules, with little to no adjustment for varying occupant behaviors. This "one-size-fits-all" approach can lead to inefficiencies and increased energy use. The study of occupant behavior in buildings is still having room for improvement. Researcher could collect large-scale data on occupant behavior in real-world settings. The insights gained from such data could then be used to develop more nuanced and effective strategies for managing energy consumption in buildings, including more flexible operation schedules for systems that are more responsive to occupant behavior. This shift could lead to significant energy savings and make our buildings more sustainable and comfortable for their occupants.

As outlined in the title of this study, our goal is to contribute to the development of theoretical foundations for building control systems that are centered around occupants, thereby facilitating a path towards truly occupant-focused control. To fully appreciate the aims of this study, it's essential to first unpack the concept of "occupant-centric control". Depending on their purpose and level of technological sophistication, buildings can house a wide variety of control devices and systems. Our primary concern at present centers around the indoor environmental control systems of buildings, which are intrinsically tied to the thermal, visual, and indoor air quality conditions that occupants experience (2020 ASHRAE Handbook). Hence, the initial interpretation of occupant-centric control is one that consistently preserves an environment that safeguards the health and comfort of the occupants. The fields of health and comfort science provide us with a broad array of insights into what constitutes ideal indoor conditions. These insights are typically formalized as a variety of specifications, standards, and guidelines, proposing measurable targets for indoor environmental conditions, such as temperature or light levels. These targets often act as the set points for control variables (ASHRAE. Thermal Environmental Conditions for Human Occupancy; 2017 Organization for Standardization; Lighting of Indoor Work Places; ISO Standard 8995:2002). Therefore, achieving these targets is the primary objective that our "occupant-centric" environmental control devices and systems can be operated to realize.

Occupant perception and interaction with building

Nonetheless, a relevant question appears - who should operate these control devices and systems? Current studies suggest that certain control systems consistently strive to keep environmental conditions near set points to maintain comfort. However, this approach invariably leads to wastage of energy and falls short of realizing "performance optimization", a term we will elucidate further below in this introduction. An alternative strategy allows each inhabitant the ability to manipulate their immediate surroundings. From the perspective of occupant autonomy, this latter approach might appear more attractive. Yet, it relies heavily on the cognitive engagement of the occupant and may necessitate repeated adjustments to achieve a comfortable environmental range. As such, this strategy fails to enhance convenience for occupants and can even be viewed as lacking in sophistication. Moreover, as this method does not leverage the building's self-perception to regulate environmental conditions, it is ill-suited for optimizing energy consumption. In light of these considerations, we wish to present a comprehensive theory focusing on how occupant's perceive and interact with their building. To this end, we envision empowering buildings with the capability to proactively perceive indoor scenarios and occupant behavior. Simultaneously, we aim to establish control methods that are both economically efficient and environmentally conscious.

Occupant privacy and comfort

Another challenge arises in the building's process of perceiving and interacting with its occupants: the preservation of occupant privacy and comfort. It is inevitable that during this perception and interaction process, the building collects data on the occupants. Pinpointing occupants' locations and behaviors may even require the use of surveillance tools like cameras and microphones, which raises significant privacy concerns and could make the occupants feel uneasy or intruded upon. Another significant contribution of our study is our approach to this delicate issue. We strive to accurately identify and differentiate various occupant behaviors using only environmental data. This approach bypasses the need for potentially intrusive tools like cameras or microphones, thus prioritizing occupant comfort and privacy.

This methodology not only addresses privacy concerns but also paves the way for more integrated, user-friendly, and ethical building control systems.

Building energy efficiency

In our point of view, "performance optimization", indicating different aspects. First aspect is the building energy efficiency. Building automation and controls, along with measurement and performance monitoring, are essential aspects of high-performance building design.

Advanced building automation and controls can optimize the operation of HVAC systems, lighting systems, and other building systems, resulting in significant energy savings. For instance, programmable thermostats can automatically adjust temperature settings based on occupancy schedules, reducing unnecessary energy consumption. Smart lighting controls can turn off lights in unoccupied areas or adjust lighting levels based on natural light availability. In addition, an optimized management of windows or doors openings can reduce the building loads (e.g. summer night's openings help decreasing or avoiding cooling). Real-time monitoring and analytics of energy use through building automation and controls allow for continuous optimization of building performance. By monitoring and analyzing building state and energy use data in real-time, building owners and operators can gain valuable insights into the performance of their buildings and identify areas of high energy consumption. This data-driven approach allows for evidence-based decision making and targeted energy-saving initiatives.

Measurement and performance monitoring begin with identifying relevant indicators, such as electricity consumption, gas usage, thermal comfort, indoor air quality, and occupancy levels, which serve as key performance metrics for quantifying building energy performance. Once the indicators are identified, energy monitoring systems with appropriate sensor combinations can be installed to track energy use in real-time. These systems collect data on energy consumption, temperature, humidity, and other relevant parameters, providing a comprehensive view of the building's performance. Advanced sensor technologies, such as Internet of Things (IoT) devices, enable remote monitoring and real-time analytics, allowing for proactive identification of energy-saving opportunities.

Buildings, whose performance is automatically monitored, analysed and potentially improved using a set of sensors and actuators are then qualified as Smart Building. These measures applied to Smart Buildings can be tailored to the specific needs and requirements of the building and its occupants. Action performed by occupants can have a significant impact on the energy consumption due to their use habits or preferences. Furthermore, not involving people in the energy performance can be counterproductive: if an action

performed automatically by an actuator is not understood, it may be blocked or cancelled by occupants. As buildings are originally design for people, it is required to take a user-centric approach to their energy efficiency optimization.

One objective of our work will be to pave the road for using user-centric optimization, and predictive control approaches, supported by building automation and controls, along with measurement and performance monitoring, to improve the energy efficiency of smart buildings. By leveraging real-time data, evidence-based decision making, and continuous performance monitoring, these approaches can lead to significant energy savings and contribute to sustainable building practices. This is our first definition for "performance optimization".

Need of accurate and diversified data

The second definition for performance optimization refers to optimal data, for which more accuracy and more diverse data can reflect to the real building indoor state. The study (Smart2B 2021) shows that poor data quality will lead to inefficiency and inaccuracy assessments and decisions. Krishana CM et al. (2023) shows that it will negatively impacts the decision-making process on time and cost, also the decision making performance in the building. Pettersen et al. (2017) says that despite having good knowledge about the properties of a building (like its insulation, windows, HVAC system, etc.), we still observe a difference between predicted and actual energy performance. This difference, or "gap", cannot be reduced without introducing larger variations in the input data models. This means, that we cannot improve the accuracy of our predictions without considering a broader range of possible conditions and variables in the models we use to make these predictions. Thus, if the input data does not accurately represent the building's conditions and the behaviors of its occupants, the model's predictions can deviate significantly from the actual energy use. Thus, the quality of data - its accuracy, completeness, and relevance - is of utmost importance to narrow the performance gap and improve the reliability of the model's predictions.

Furthermore, it implies the necessity of incorporating a broader range of conditions and variables into the data models. This suggests that not only the quality but also the diversity and comprehensiveness of data can impact the accuracy of energy performance predictions. However, optimal data is gathered from the sensors, placing sensors into different location will lead to a totally different result, thus, to find the optimal sensor

location is crucial for the data. In the thesis, another contribution of ours is determining the optimal placement of sensors in buildings in order to obtain optimal data.

Overview of the thesis

Based on the above mentioned challenges, the following points will be developed in the thesis. The first chapter is dedicated to the presentation of the main scientific barriers through a review of the existing bibliographic literature (Chapter 1), and a second chapter is introducing the thesis case studies (Chapter 2).

In Chapter 3, the optimal sensor location is investigated, in order to guarantee the data accuracy for building performance. Determining the optimal sensor placement, that appropriately describe the building state and the user comfort, requires considering various factors, such as building layout, sensor coverage, cost-effectiveness, and the targeted activities or usages to be monitored. Finding optimal solutions for sensor placement is a challenging task that involves interdisciplinary research combining building physics, sensor technologies, data analytics, and optimization techniques.

In Chapter 4, a method for accurate detection of activity and occupancy in a building or a room is proposed. Our primary contribution is the accurate detection of occupant behaviors using machine learning algorithms, considering significant constraints such as occupant privacy and the presence of unlabeled data. In addition, we also address common challenges like imbalanced data between normal and abnormal operating conditions, creating realistic scenarios of activities and usages, and developing robust and scalable real-time activity detection and analysis techniques. A benchmark for different machine learning algorithms has been made for comparing performance in order to select a more proper algorithm. The significance of real-time detection can't be overstated, given the quick and dynamic changes that can occur in indoor environments. Thus, the speed of detection becomes a key factor. In our benchmark testing, we will compare the detection speed of various algorithms and implement an alternative strategy, which remarkably reduced the time required for detection. With these contributions, we can ensure reliable and accurate activity and occupancy detection in smart buildings, which forms the foundation for effective energy management strategies and personalized user experiences.

Another significant challenge in smart buildings is the detection of multiple activities occurring simultaneously (Chapter 5). This requires developing sophisticated algorithms and techniques that can differentiate between overlapping activities, recognize complex

patterns of activities, and handle the dynamic nature of activities in real-time. The detection of multiple activities presents unique challenges, including the need for advanced sensor fusion techniques, robust and adaptive algorithms, and efficient data processing methods. Addressing these challenges would enable smart buildings to gain a holistic understanding of the diverse activities and usages of their occupants, leading to more effective and personalized energy management strategies that account for the dynamic and complex nature of human activities.

As this thesis delves into these scientific challenges and issues, it aims to contribute to the advancement of the field of smart buildings and pave the way for more energy-efficient and sustainable building practices. By addressing these challenges, we can unlock the full potential of user-centric optimization and predictive control approaches in smart buildings, leading to more intelligent, responsive, and sustainable built environments.

LITERATURE REVIEW

Energy management in smart buildings involves various aspects, including accurate measurement of building performance, reliable data collection, data analytics for prediction and detection, and user-centric approaches. This chapter provides a comprehensive review of the existing literature in these areas, highlighting the key concepts, challenges, and research gaps. The first part 1.1 of the literature review focuses on the measurement of energy performance and reliable data collection in buildings. Energy performance indicators are first presented as important metrics for evaluating the energy efficiency of buildings, highlighting what needs to be measured. Different types of sensors used for data collection are then discussed, including their functionalities, applications, and limitations, emphasizing how to measure. Furthermore, the issue of the Optimal Sensor Placement (OSP), which refers to the strategic deployment of sensors to optimize data collection and analysis, is introduced as a critical challenge in the field. The second part 1.2 of the literature review shifts the focus towards the occupants of the building, recognizing their significant influence on building performance. This part emphasizes the concept of a user-centric approach, also known as "human in the loop", which considers occupants as virtual sensors who can provide valuable data and information. The role of occupants in decision-making processes related to energy management is also highlighted. The third part 1.3 of the literature review focuses on methods for detecting and identifying activities and events in buildings, particularly machine learning classification- and regression-based approaches. This part centers on data analytics and prediction of building performance based on the activities identified in the previous part. It explores techniques for analyzing data and predicting building performance, using the information obtained from occupant behavior detection and activity identification. Lastly, the chapter concludes in section 1.4 with an overview of model predictive control, a key approach used in user-centric energy management to optimize building performance based on predicted outcomes.

1.1 Performance monitoring of buildings

Three main key levers have been presented in the general introduction to improve the energy performance of buildings: sobriety, efficiency, and the use of renewable energy. While the first lever, sobriety, emphasizes reducing energy consumption through behavioral changes and lifestyle choices, and the second lever, efficiency, seeks to optimize the energy use within buildings, the third lever, renewable energy utilization, promotes the use of sustainable energy sources. This lever encompasses various aspects, including building envelope design, HVAC systems, lighting, and appliances, among others. In the context of efficiency, measurement and monitoring of building performance play a critical role. Understanding the energy performance of a building is fundamental to identify areas of improvement, validating design choices, and evaluating the effectiveness of energy-saving measures. Therefore, the first section of this literature review chapter focuses on the measurement and monitoring of building performance. Specifically, it addresses two key problems. First, what defines/qualifies the energy performance of a building? This question explores the concept of energy performance, which encompasses various aspects such as energy consumption, energy needs, energy demand, energy efficiency, thermal comfort, indoor air quality, and carbon footprint. It aims to provide a clear understanding of the many aspects of building performance and the different dimensions that need to be considered. The second problem to deal with is how building performance is measured? This question explores the methods and techniques used for measuring and monitoring building performance. It encompasses both quantitative and qualitative approaches, ranging from analyzing conventional utility bill and collecting data from sensors to advanced modeling and simulation techniques. It also discusses the challenges and limitations associated with different measurement methods, such as accuracy, reliability, scalability, and cost-effectiveness. By addressing these problems, this section of the literature review provides a comprehensive overview of the concepts and methods related to the measurement and monitoring of building performance. It establishes the foundation for the subsequent sections of this chapter, which will explore the state-of-the-art research and practices in the fields of data analytics, and decision-making approaches for improving building energy performance.

1.1.1 Energy performance indicators

In the scientific community, various building Energy Performance Indicators (EnPIs) are used to monitor and measure energy consumption in buildings. These indicators can be categorized into five main categories:

- Energy Consumption: This indicator refers to the total amount of energy utilized by a building or a system, measured in kilowatt-hours (kWh). The Energy Use Index (EUI) (Fairey and Goldstein, 2016; Hsien-te and Chia-ju, 2021), belongs to this category.
- Energy Needs: This indicator pertains to the amount of energy required to meet the demands of a building or a system. The Energy Conservative Index (ECI) is an example of energy-needs indicators (Lon et al. 2014).
- Energy Demand: This indicator denotes the instantaneous or peak power demand of a building or a system, measured in kilowatts (kW). The Energy Demand Intensity (EDI) and the Peak Energy Demand (PED) are two examples of energy demand indicators.
- Energy Efficiency: This indicator signifies the ratio of energy output to energy input, usually expressed as a percentage or a decimal. The Energy Efficiency Index (EEI), the Building Energy Index (BEI), the Home Energy Rating System (HERS) and the zero Energy Performance Index (zEPI) are four already used energy efficiency indicators (Hayati et al. 2014; Moghimi et al. 2013).
- Carbon Footprint: This indicator quantifies the amount of carbon dioxide (CO₂) emissions associated with energy consumption, measured in metric tons of CO₂ or CO₂ equivalent.

It is worth noting that these EnPIs are commonly used in the scientific and technical literature to assess the energy performance of buildings. They provide essential metrics for evaluating energy consumption, demand, efficiency, and environmental impact, enabling researchers and practitioners to better understand and quantify the energy performance of buildings.

Some of the above-mentioned indicators (EUI, ECI, EDI, EEI, BEI, HERS, zEPI and CO₂ indicator) are presented in Annex A.1

Several studies have explored the relationship between building energy performance and energy needs. Using energy modeling or simulation software, researchers have estimated the energy needs of buildings and compared them with their actual energy consumption. Incorporating energy needs as an indicator of building energy performance

provides a more accurate and comprehensive assessment of energy performance, as it is an estimation of the amount of energy needed to meet the specific requirements of a building in terms of heating, cooling, lighting, etc. (Li et al. 2022). If incorporating energy needs as an indicator of building energy performance offers advantages in terms of accuracy and comprehensiveness, challenges such as data quality and model complexity need to be addressed to ensure reliable and consistent assessment of building energy performance using energy demand-based indicators. Both concepts of “energy needs” and “energy demand” are related, but the first is a theoretical estimation based on modeling or simulation, while the latter is the actual measurement of energy consumed. It is the measurement of the energy used to meet the energy needs of a building, as measured by energy meters. An example is the Peak Energy Demand (PED) measuring the peak energy demand of a building, which refers to the highest amount of energy required by the building at any given time. It is often used to assess the capacity of a building’s energy systems and to identify potential areas for energy demand management. Comparative analyses of building energy performance indicators based on energy-demand have been conducted by Behzad et al. (2019). According to the authors, energy demand-based indicators can provide a more nuanced understanding of building energy performance, capturing the dynamic nature of energy requirements in buildings.

By integrating an emissions indicator, such as CO₂ emissions, into performance metrics such as EUI, Energy Cost Index, and EDI, a more comprehensive assessment of a building’s environmental performance can be obtained. This allows for a more holistic approach to evaluate the sustainability of a building, considering both energy performance and carbon emissions. By incorporating an emissions indicator, building owners, operators, and stakeholders can better understand the impact of their building on the environment and make informed decisions to optimize energy performance and reduce carbon emissions (IEA, 2015). This approach should be coupled to an estimation of the CO₂ emissions of the other life cycle stages (building construction, renovation and end-of-life) for a more holistic view of the impact of buildings on climate change.

Among all possible indicators, energy consumption is used as the primary focus in this research. Specifically, the total heat load is set as the indicator for this study, which is objective and easy to understand without the need for reference. Choosing a classic indicator such as ‘Energy Consumption’ has several compelling reasons. Firstly, Energy Consumption is a widely used and recognized indicator in the scientific community for monitoring and measuring energy performance in buildings. It provides a straightforward

and quantifiable measure of the total amount of energy utilized by a building or system, typically measured in kilowatt-hours (kWh). Secondly, Energy Consumption is easy to measure and obtain through energy meters, making it a practical and accessible indicator for research and analysis. Thirdly, Energy Consumption reflects the actual energy used by a building to meet its energy needs, which makes it a relevant and meaningful indicator in evaluating the performance of energy systems and identifying areas for improvement. Furthermore, Energy Consumption can serve as a benchmark for comparing buildings or systems, allowing for meaningful comparisons and assessments of relative performance.

To measure this indicator, an energy monitoring system is necessary, which requires the use of sensors to collect data.

1.1.2 Monitoring and sensors

With the importance of energy consumption as a main indicator established, the next step in our research is to explore the context of energy monitoring systems and sensors for building performance measurement. The energy performance of buildings must not solely be determined by the quality of materials, technical systems, and control strategies, but also by the behavior of their occupants. Thus, whenever the notions of building performance monitoring are mentioned in the remainder of this thesis report, it should be considered that this systematically encompasses the influence of the occupants because they have a strong impact on the building performance as it will be discussed in section 1.2.

In recent years, advancements in technology have led to advanced systems that can collect, analyze, and understand how a building uses energy. These systems use a wide range of sensors, meters, and devices to capture real-time data on energy consumption, environmental conditions, and other relevant parameters. In particular, smart buildings have emerged as a promising frontier in building performance evaluation, where the integration of cutting-edge technologies enables intelligent management and optimization of energy use. In this section, we will examine how energy monitoring systems work, the types of sensors used for measuring building performance and detecting activities/events, and the challenges associated with smart buildings aiming for energy efficiency and sustainability.

1.1.2.1 Energy monitoring or management systems

Energy management systems (EMS) allows homeowners to monitor and control their energy usage effectively. EMS offers several advantages, starting, firstly, with reduced electricity bills by reducing unnecessary or excessive appliance usage. Secondly, it helps in lowering greenhouse gas emissions and reducing reliance on fossil fuels, thus contributing to environmental sustainability. Thirdly, it provides homeowners with real-time feedback and insights on their energy usage through smart devices and apps, allowing them with increased control and awareness.

EMSs comprise various components and functionalities. It connects diverse home appliances, distributed power sources, and energy storage, along with other equipment, through advanced metering infrastructure (AMI), facilitating monitoring and management of electrical equipment. Ma et al. (2021) provides a comprehensive overview of the structure, functions, and key technologies of home energy management systems (HEMS). The study also analyzes the control strategies of HEMS and highlights how electricity market reforms have created favorable conditions for incorporating demand-side load resources into supply-demand regulation. The increasing electrification of residential properties makes residential load resources a valuable asset for demand-side load regulation, as resident home appliances participate in the "two-way interaction" with the power grid in the form of Demand response (DR).

The relationship between energy efficiency and digitalization has been studied in reports from the International Energy Agency (IEA, 2017). Digital technologies can significantly improve energy efficiency in areas like transportation, buildings, and industry. They allow to optimize energy use and enhance connectivity between devices and machines, making the entire energy system more efficient. However, the growing number of devices and data servers from digitalization could also lead to higher energy consumption if not managed carefully. Policymakers face the challenge of guiding digitalization in a way that maximizes its benefits for the energy system while minimizing negative impacts. Their energy saving strategies necessitate the integration of users and appliances, as it will be detailed in the section 1.2.1 on the Influence of occupants' behavior.

1.1.2.2 Sensors for measuring building performance

With the development of EMS and smart houses, numerous sensor types have been developed. Sensors can be classified into four main categories, as mentioned in the article

by Bae et al. (2021), namely environmental sensors, occupancy sensors, motion and vision-based sensors, and energy consumption sensors.

The first category, the environmental sensors, including traditional HVAC sensors, play a crucial role in monitoring and optimizing the performance of building HVAC systems. This kind of systems can range from packaged units to built-up systems, and commonly used equipment including chillers, boilers, heat pumps, fans, pumps, valves, heat exchangers, filters, dampers, diffusers, ducts, and pipes. HVAC sensors are used to monitor various parameters such as temperature (e.g., outside air temperature, chilled water temperature, supply air temperature), humidity, flow (e.g., chilled water flow rate, supply airflow rate), pressure (e.g., chiller water pressure, duct static pressure), and gas flow for absorption chillers or boilers. These sensors are used in the control loop that modulates the equipment to maintain the controlled variable (e.g., temperature or pressure) at a set point. For example, in a Variable Air Volume (VAV) system, the supply air temperature set point is maintained by modulating chilled water and hot water flow, and the zone temperature set point is maintained by modulating the VAV box damper position or turning on reheating. HVAC equipment is often equipped with electrical meters, but additional sensors such as BTU (British Thermal Units) meters may be needed to establish baseline energy consumption and verify the improved performance enabled by advanced control methods (Narayanan et al. 2012).

The second mentioned category is composed of the occupancy sensors which are used to detect the presence and number of people in a room or specific area of a building, using technologies such as motion sensors, heat body sensors, optical presence sensors, etc. Information on occupant presence can be used to control lighting and HVAC systems, as well as adjust temperature set points. The number of occupants can be used to modify ventilation rates, and this information can also be used to estimate and predict various building loads for optimal HVAC system control. Finally, occupant identity and location can potentially be used to provide customized indoor environments to maximize occupant satisfaction while optimizing energy efficiency. Previous studies have suggested that adopting occupancy-based control, which deploys energy where demand exists, could significantly improve building energy efficiency without compromising occupant comfort (Brooks et al. 2015). Occupancy-based control for lighting systems has been relatively straightforward to implement due to their instantaneous response. Various building standards and green building rating programs have recommended occupancy-based lighting control, with reported energy-savings potential ranging from 20% to 75% (Delaney et

al. 2009). However, existing occupancy-sensing technologies may occasionally cause false on/off switches of lighting systems, leading to occupant dissatisfaction and energy waste (Guo et al. 2010). With the success of occupancy-based lighting control, efforts have been made to incorporate occupancy information into HVAC controls. Yet, due to the slower response time of HVAC systems, predictive strategies are often needed to ensure occupant comfort and minimize energy consumption, making practical and scalable occupancy-based HVAC control more challenging compared to lighting control. Variations in ambient conditions, such as air temperature, humidity, CO₂ levels, acoustics, and light, can also be used to infer occupancy information. One advantage of methods based on ambient condition variations is that they can potentially infer the number of occupants in a space. These methods rely on the correlation between ambient condition changes and the number of occupants and utilize differential equations or statistical/optimization-based mapping methods, such as regression or machine learning algorithms, to establish the relationship. However, adjusting or tuning these equations or methods for different spaces and buildings may require additional costs for transferability (Chen et al. 2018). Among the methods in this category, those using variations in CO₂ concentration are the most common, although the slow gas mixture in buildings may limit the estimation performance of methods relying solely on CO₂ concentration (Meyn et al. 2009).

Motion sensors, such as passive infrared (PIR), ultrasonic, and microwave sensors, complete this category of occupancy sensors. They have been widely used due to their low cost, low power consumption, small form factor, nonintrusive nature, and privacy-preserving characteristics. PIR sensors, in particular, are popular in real-world applications due to their robustness to environmental variations (Luo et al. 2016). However, relying solely on motion sensors has significant limitations (Guo et al. 2010). These sensors can only provide information about occupant presence or absence, and more detailed information such as the number of occupants, identity, and location is typically unavailable. Moreover, motion sensors require detectable movements to trigger, and if no movement occurs, a delay of some minutes is usually applied before switching from the occupied to unoccupied state, resulting in potential energy waste. Additionally, motion sensors can be triggered by other objects, such as hot beverages, appliances, or pets, and PIR sensors require a direct line of sight between the sensor and occupants in a space.

In recent years, a third category of sensors, based on vision analysis has gained attention due to rapid advancements in sensing, computing, and computer vision technologies. These methods utilize occupants' interactions with furniture, such as chairs and doors

(Kim et al. 2019a), appliances like computers, communication networks such as Wi-Fi and Bluetooth (Zou et al. 2018a), and corresponding plug loads (De Coninck and Helsen, 2016), to estimate occupancy information. Vision-based sensors offer the potential for more detailed and accurate occupancy estimation compared to motion sensors, as they can capture visual cues beyond just motion. However, vision-based methods may also raise concerns related to privacy and data security, as they involve visual data capture and processing. Thus, careful consideration of privacy protection measures should be considered when implementing vision-based sensors for occupancy estimation.

The fourth and last category proposed by Bae et al. (2021) includes the energy consumption sensors. This category integrates power meters playing an essential role in modern commercial buildings for monitoring and managing heat consumption and electricity usage. These sensors enable building-level or submetering level measurement of energy consumption, providing valuable insights into the factors influencing energy usage. By analyzing energy or power metering data, opportunities for energy efficiency improvements can be identified, and control methods can be explored to lower energy demand and cost. Power measurement data are also critical for developing and validating power consumption models used in predictive control strategies. Additionally, metering data can provide direct feedback for advanced control techniques, such as extremum seeking control, which systematically identifies control inputs that minimize a measured objective function, such as power consumption. Advancements in two-way communication and power metering technologies, such as one-way automatic meter reading and bidirectional smart meters, are enabling autonomous responses of building automation systems and other connected devices to dynamic electricity prices and grid signals. This allows for more precise and efficient demand response strategies, where buildings and devices can adjust their energy usage in response to grid conditions.

In the near future of advanced building controls, smart Internet of Things (IoT) sensors are emerging as a promising solution. These sensors have wide-ranging applications in various aspects of building energy management systems, including demand-controlled ventilation, energy recovery ventilation, outdoor air systems, CO₂ sensing, ultraviolet germicidal irradiation, displacement ventilation, and underfloor air distribution (Minoli et al. 2017). Additionally, GPS-based remote sensing capabilities are expected to become ubiquitous, adding a layer of flexibility to building energy control schemes. IoT sensors are also being integrated with advanced control methodologies, leading to projected energy cost savings of 30-40% and reduced peak demand power. For example, Tran et al. (2019a)

connected IoT sensors to lighting, plug loads, and HVAC systems, transferring data to the cloud for scheduling optimization. Moreover, OSP strategies have been proposed by the same first co-author (Tran et al. 2019b) to maximize sensing and observability with minimal infrastructure within an IoT framework.

As we conclude this section about sensors, it is also important to highlight the significance of their location in building control loops. Buildings are typically composed of multiple thermal zones, each with unique temperature profiles influenced by factors such as occupant behaviors or solar radiation. Even within the same thermal zone, temperature profiles may vary across different areas. but, many engineering designs and practices often involve selecting a single sensor location based on expert knowledge and room configuration, rather than installing multiple sensors due to concerns over construction and maintenance costs. We will focus, in section 1.1.3, on the importance of sensor placement and how it can impact the accuracy and effectiveness of building control strategies and explore further and try to propose a solution about this point in Chapter 3.

1.1.2.3 Smart home technologies

A smart home, also known as a connected home or smart house, is a residential space that incorporates advanced technology to provide automation, convenience, and control of various household systems and appliances, which can greatly contribute to reduce energy demand (Alam et al. 2012) (Balta-Ozkan et al. 2014). The concept of smart homes has gained popularity in recent years, driven by advances in IoT technology and the availability of affordable and easy-to-use devices (Marikyan et al. 2019). To achieve energy conservation in a smart home, various components must work together effectively. Figure 1.1 provides an overview of a smart home energy management system with key modules. This system includes a smart automation console that communicates with smart plugs connected to different devices in the home, such as appliances, lighting, and door locks. The console controls the functionality of all connected devices, profiles energy consumption details, and monitors energy demand. It also communicates with smart grids through AMI to provide energy demand requirements and reads energy price data from energy suppliers via the Internet. Using this pricing data, the smart automation console presents the user with energy-saving options by giving a complete overview of various energy-consuming appliances. The user can then choose to switch off or schedule appliances with high energy usage to low-price hours.

Technologies enable the connection and control of multiple appliances through wired,

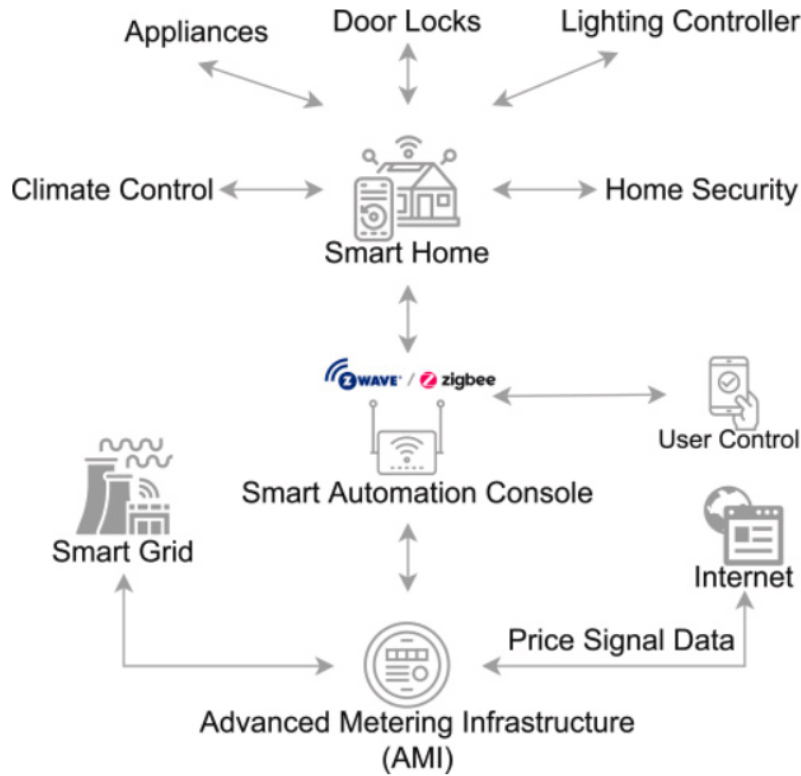


Figure 1.1 – Description of Advanced Metering Infrastructure (AMI)

wireless, or hybrid mediums. Variety of these technologies are commonly discussed in the literature (Alhamoud et al. 2014) (Hafeez et al. 2020) including ZigBee, Z-wave, X10, WiFi, Bluetooth, Radio Frequency (RF), Radio Frequency Identification (RFID), PowerLine Communication (PLC), Ethernet, KNX, LoRaWan. These technologies provide users with interfaces for intelligent control of all smart home appliances each offering different functionality options. They communicate with different smart appliances via wired or wireless channels to record usage time and energy consumption data. The pros and cons of different technologies are summarized in the Table A.2 in Appendix A.

Upon analyzing the advantages and disadvantages of different smart home technologies, it becomes clear that wireless technologies offer easy installation and efficient management of multiple devices. However, the security risks associated with wireless technologies are significantly higher than those of wired technologies. On the other hand, wired technologies can be challenging to install, particularly in old buildings where the existing communication wiring structure may not be available.

In this thesis, in the context of occupancy estimation, where reliable data collection, storage, and transfer are crucial, the choice of technology plays a significant role. After careful consideration and evaluation of various options, we have determined that WiFi is the most suitable technology for our needs. WiFi offers several distinct advantages over other competing technologies. Firstly, WiFi provides a widely adopted and standardized communication protocol, ensuring compatibility and interoperability with a wide range of devices and systems. This compatibility facilitates seamless integration into existing infrastructures, reducing implementation complexities and costs. Secondly, WiFi offers a higher data transfer rate compared to technologies like Bluetooth, ZigBee, or LoRaWAN, enabling faster and more efficient transmission of occupancy-related information. Additionally, WiFi networks are known for their robustness and stability, offering reliable data collection and minimizing the risk of signal disruptions. Moreover, WiFi provides sufficient coverage for most indoor environments, eliminating the need for extensive deployment of additional access points or devices. Lastly, WiFi infrastructure is readily available in many buildings and public spaces, making it a cost-effective solution that uses existing infrastructure.

1.1.3 The Optimal Sensor Placement (OSP) issue

As introduced at the end of section 1.1.2.2, sensor placement is an important factor in control loops. Most engineering designs and practices pick one location arbitrarily, or based on expert knowledge or constraint in the room layout, rather than installing multiple sensors because adding more sensors will increase the construction costs and maintenance costs. Finding the OSP is an important aspect of many scientific, engineering, and industrial fields and involves a variety of applications, including (structural) health monitoring, environmental monitoring, and many others. The goal is to determine the best positions for sensors to collect accurate and reliable data in target environments or situations. This OSP problem has been deepened in recent years along with advances in sensing technologies, which have enabled the collection of vast amounts of data from complex systems.

Determining the OSP is not a simple task; it involves trade-offs between several conflicting objectives, such as data accuracy, the number of sensors required, and the cost of sensor deployment. In many cases, we want to reduce data uncertainty while achieving the most accurate results possible. This requires careful consideration of the measurement process, the environment, and the desired goals. In the following sub-sections, after

a brief state-of-the-art of the different applications and methods found in the literature, disadvantages and advantages of different methods are pointed out.

1.1.3.1 OSP problem in literature from various domains

As mentioned above, the search for OSP is a topic of interest in various applications, particularly in the fields of structural health monitoring (Yi et al. 2011, Kammer, 1991), geological disaster detection (Dong et al.2018 and Limongelli et al. 2003), wireless networks (Castello et al. 2010), and building performance (Yoganathan et al. 2018, Tian et al. 2018 and Bianco et al. 2012). For example, in bridge monitoring, sensors are strategically placed to detect changes in the structural integrity of the bridge, such as cracks or deformations (Stubbes et al.1996 and Heo et al.1997). By analyzing the data collected by these sensors, the health of the bridge can be assessed more accurately. Similarly, in building structural monitoring, OSP can improve the accuracy of data collected on changes in temperature, humidity, and vibration, leading to more accurate assessments of building health.

OSP is also of paramount importance for the energy performance of buildings. Most of the methods developed in the context of smart buildings aim to find the optimal placement of sensors to improve the monitoring and control of buildings. This requires careful consideration of significant factors that directly affect the building's performance. Previous studies, such those of Tian et al. (2018), Yoganathan et al. (2018) or Suryanarayana et al. (2021) have shown that sensor locations and types can have a significant impact on building performance and HVAC system performance, respectively. For example, changing sensor positions in one of a building's rooms resulted in an HVAC energy variation between -0.34% and 0.14% (Du et al. 2015).

The zone air temperature, controlled by thermostats, is a commonly used parameter for controlling the indoor thermal environment in buildings. However, simulation and experimental studies have shown that the temperature profile along the vertical direction at different locations in a building can be non-uniform and stratified, and the temperature profile can also vary under different conditions such as supply airflow rate, geometry of the zone, climate region, and seasonal variation (Du et al. 2015) (Yoganathan et al. 2018; Shan et al. 2019). Therefore, thermostat locations significantly affect building energy usage and occupants' thermal comfort.

In the field of modern building automation systems, wireless sensor/actuator networks are commonly used for lighting controls. A codesigned strategy was applied by Maasoumy

et al. (2013) for building lighting controls, resulting in 45% energy savings and a 23% reduction in sensor costs. Another study from Kim et al. (2020) utilized low-cost sensor networks for lighting glare controls. But challenges such as simultaneous monitoring of numerous variables, data management, power consumption of wireless sensors, and maintenance issues remain.

1.1.3.2 Approaches and methods for addressing OSP problem

In the previous sub-section, the state of the art discussed the topic of OSP from an application perspective. When it comes to determining where and how many sensors should be placed in buildings, there are two main approaches. The first approach relies solely on the expertise of domain experts, while the second combines expert knowledge with analytical or numerical methods.

Expertise-based methods involve using the knowledge and experience of domain experts to determine optimal sensor locations. These methods are based on guidelines, best practices, and heuristics that are developed based on expert knowledge of the system being monitored. These guidelines for good practices often establish to avoid placing sensors in locations where there is a risk of measuring inaccurate or unreliable data (Frei et al. 2021). For example, in the case of temperature sensors, it is well-known that direct exposure to sunlight can cause measurement errors. Therefore, guidelines may recommend placing sensors in shaded areas to obtain accurate temperature measurements. Similarly, guidelines may also recommend avoiding sensor placement in areas with high turbulence or flow disturbances, as these can affect the accuracy of measurements. Another approach to expertise-based sensor placement is the trial-and-error method. This method involves initially placing sensors in specific locations and then adjusting their locations based on observations of unexpected phenomena affecting the measurements after a certain period of time. For example, if unexpected changes in measurement data are observed, sensors can be moved to different locations to obtain more reliable measurements. This trial-and-error approach relies on the experience and intuition of domain experts who can identify potential sources of measurement errors and take corrective actions accordingly.

Expertise-based sensor placement methods can be particularly useful in cases where analytical or numerical methods may not be feasible or practical due to limitations in data availability, computational resources, or system complexity. Huang et al. (2014) proposed a method based on the expert method supplementing Computational Fluid Dynamics (CFD) simulations to control indoor temperature and improve efficiency in a

large-scale office building. This method is simple and practical, but there are no criteria to evaluate its effectiveness. However, these methods leverage the knowledge and experience of domain experts to guide sensor placement decisions and can provide valuable insights for optimizing sensor placement in real-world applications.

As the above-mentioned study by Huang et al. (2014) shows, methods based on expert knowledge can be complementary to analytical or numerical methods. The latter methods can themselves be the sole basis for OSP approaches.

On one hand, analytical methods for OSP are based on mathematical and statistical models that are used to determine the optimal sensor locations. These methods are generally fast and efficient, but they are limited by the assumptions made about the system being monitored and the measurement process. They rely on mathematical models and statistical techniques to estimate optimal sensor locations based on minimizing measurement uncertainty or enhancing the accuracy of the parameter estimate of an analytical model. Chen and Li (2016) used Bayesian optimization to develop virtual sensors by combining prior knowledge of temperature statistics and Bayesian model fusion to predict spatial temperature distribution, aiming to reduce the number of required sensors.

Maximum entropy is another analytical method for sensor placement that aims to minimize measurement uncertainty. It is based on the principle that when you lack other information, the most unbiased probability distribution is the one with maximum entropy. This method estimates the probability distribution of the system being monitored and determine optimal sensor locations that maximize entropy, thus minimizing uncertainty in the measurements. Maximum entropy methods can also be seen as information-theory-based methods because they select sensor locations that provide the most informative data to improve understanding of the system being monitored. Papadopoulou et al. (2016) proposed a method involving three sequential sensor placement strategies based on information entropy analysis to find the optimal sensor location for outdoor wind speed detection. This method has advantages, such as considering a time-dependent system and modeling error. However, this method does not consider systemic errors and spatial correlations between errors, and it assumes constant modeling error. Although interesting, the two above-mentioned methods (Bayesian Optimization, Maximum Entropy) are very rarely used for sensor location for building performance monitoring. At the time of writing, only applications in structural health monitoring (SHM) of buildings seem to have been studied (Osegueda et al.1997).

This is also the case for methods based on Maximum Likelihood. They are very often

cited as a potential candidate (Faridah et al. 2021) method but are still confined to the field of SHM or geotechnics. Maximum likelihood is an analytical method for sensor placement that is based on estimating the parameters of an analytical model. Analytical methods aim to maximize the accuracy of parameter estimates in a mathematical model of the system being monitored. These methods select sensor locations that are most likely to provide accurate estimates of the parameters of interest, thus minimizing measurement uncertainty. They may involve optimizing the sensor locations based on the trade-off between the cost of sensor installation and the expected benefits in terms of improved measurement accuracy or system understanding.

On the other hand, numerical methods for OSP are based on simulation and optimization techniques that are used to find the optimal sensor locations. One category of numerical methods for sensor placement involves the use of optimization methods, such as Genetic Algorithm (GA) (Hou et al., 2018) (Villa et al., 2022) and Particle Swarm Optimization (PSO) (Li et al., 2022a) (Hassani and Dackermann, 2023). These optimization-based methods involve generating a set of candidate sensor locations and iteratively optimizing a performance metric, such as measurement uncertainty or information gain, to identify the optimal sensor locations. Löhner and Camelli (2005) tried to find the optimal sensor location using GA for detecting contaminant gas emissions based on CFD simulation results for a residential block and a shopping mall, but no experiments supported the simulations. Tian et al. (2018) proposed an optimization platform that uses PSO algorithm as the optimization engine to seek the optimal placement of thermostats in an office room with displacement ventilation and a Variable Air Volume (VAV) terminal box, achieving either the best thermal comfort or the least energy consumption.

Another category of numerical methods for sensor placement involves adapted clustering methods, such as k-means clustering (Kalluri, 2017) (Gautam et al., 2022). Clustering methods involve grouping data points into clusters based on their similarity, and these methods can be adapted for sensor placement by clustering potential sensor locations based on their spatial characteristics, system dynamics, or other relevant factors. Yoganathan et al. (2018) proposed a data-driven method that includes such a clustering algorithm and the Pareto principle to select the optimal sensor measurement points in an office building. The authors used the k-means clustering method to divide office spaces

into homogeneous groups based on their thermal behavior. Then, they used these groups to determine the optimal locations of temperature sensors to maximize the accuracy of building energy consumption prediction models. On another front, Shimosaka and Saisho (2016) proposed a statistical method using Received Signal Strength Indicator (RSSI) sensors to detect occupants. The optimal sensor locations are chosen using a probabilistic multi-task classification by considering, for example, financial and privacy constraints.

Compared to analytical methods, numerical approaches for OSP are often more versatile, accurate, and capable of handling realistic scenarios compared to analytical approaches. However, they may also require more computational resources and time, and may involve more complex implementation. The choice between analytical and numerical approaches for OSP depends on the specific problem requirements, available resources, and desired level of accuracy and complexity.

In Table A.2 (In Appendix A), we have included a common method for selecting the best sensor locations, along with its advantages and disadvantages. For this thesis, we aimed to find a method that combines the strengths of both analytical and numerical approaches. Specifically, we sought a method that is straightforward, efficient, precise, adaptable, and capable of optimizing multiple objectives simultaneously. More details can be found in Chapter 3.

1.1.4 Other Challenges in smart building

1.1.4.1 The sensor accuracy challenge

Sensors are typically calibrated by manufacturers to ensure their accuracy. However, sensors can still exhibit low fidelity due to various factors such as harsh environments or manufacturing defects, resulting in sensor faults. HVAC systems can experience different types of sensor faults, including precision faults that affect the precision of sensor readings due to measuring noise, and bias faults that result in sensor readings deviating from the true reading. Figure 1.2 illustrates a typical diagram of precision and bias in sensor faults. Sensor faults, also known as incipient faults, can evolve over time and may change slowly, potentially going unnoticed and affecting control performance. Despite this, there are limited studies that focus on incipient faults, with one study proposing a methodology to estimate incipient fault magnitude for HVAC systems (Zhou and Dexter, 2009).

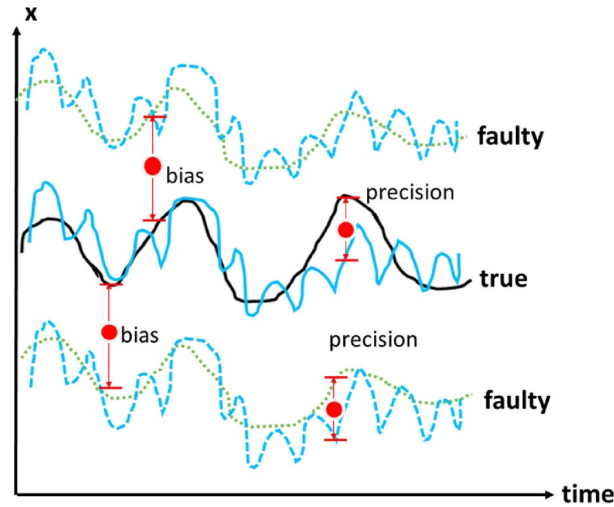


Figure 1.2 – Sensor faults and classification (Bae et al., 2021)

Multiple studies have investigated sensor errors and their impact on sensor fault distributions. For instance, normal distributions have been applied to model sensor errors in a study that focused on fault impact analysis framework (Li and O’Neill, 2019). Another study investigated the impact of sensor faults on building energy consumption for demand control ventilation, also using normal distributions for sensor errors (Lu et al., 2020a). As different sensor types may exhibit different error characteristics, it is important to consider the specific error characteristics of each sensor type.

In our thesis, the issue of maintaining sensor accuracy and preventing measurement biases was a critical consideration. We took several approaches to address this challenge. Firstly, certain sensors, such as those integrated into the weather station, underwent meticulous factory calibration to ensure high measurement precision. Secondly, for sensors involved in volatile organic compound (VOC) detection, we used the advantage of self-calibration through self-referencing. These sensors utilized an internal reference material with well-established properties to continually verify and adjust measurements in real-time. Additionally, specific calibration protocols were meticulously followed for other sensor types. As an example, for CO₂ sensors, the environment in which the sensors were deployed was carefully regulated. The room was opened to the external atmosphere until a consistent measurement level was achieved across all installed sensors. Manual adjustments, including gains, thresholds, and offsets, were then performed to fine-tune the sensor readings. An important aspect ensuring measurement accuracy was the presence of redundancies within the sensor network. This redundancy allowed us to identify and isolate

any faulty sensors within the system. Detailed information regarding these calibration procedures can be found in Section 3 of Chapter 2.

1.1.4.2 The data privacy and user acceptance challenge

In addition to energy conservation, security and privacy remain substantial challenges in the smart home market.

Ensuring data privacy in smart homes is challenging due to issues like transparency, control, and concerns about breaches, hacking, and misuse (Guhr et al., 2020). The Connectivity Standards Alliance (CSA) is addressing this with a Data Privacy Working Group (Pattison Tuohy, 2023). They aim to provide clear information on data usage and certify compliant devices. This is especially significant in the absence of comprehensive federal data protection laws in the US.

Additionally, understanding how users perceive and adopt smart home tech is crucial. Mashal et al. (2020) explored this, considering trust, awareness, enjoyment, risks, usefulness, and ease of use. Their findings emphasize the importance of these factors on users' attitudes and intentions toward smart homes.

In this thesis, we addressed the challenges of data privacy and user acceptance concerning the use of measurement devices for assessing energy performance and monitoring their usage. To ensure privacy and enhance acceptance, we carefully selected a range of sensors, excluding cameras, that respect user privacy. For instance, our sound sensors do not record conversations, and all data collected is transmitted via secure WiFi to the University of Angers' secure server (UACO₂). This approach ensures that personal conversations are not captured, and data transmission is safeguarded. By leveraging these measures, we aimed to strike a balance between gathering valuable energy performance insights, respecting user privacy, and maintaining a high level of data security.

1.1.5 Intermediate conclusion

The first section of Chapter 1 has provided an overview of various concepts related to performance monitoring of buildings. This includes energy performance indicators, monitoring and sensors, OSP issue, and other challenges in smart buildings such as sensor accuracy, data privacy, and user acceptance. In the section 1.2 of this chapter, the focus will shift towards the role of occupants in smart buildings. Occupants are not just passive users of smart home technologies, but they can also act as virtual sensors, providing data

and information that can influence building performance and decision-making processes. The concept of "User centric approach" or "human in the loop" will be highlighted, showcasing how occupants can actively contribute to the operation and management of smart buildings.

1.2 Occupants' activities, occupancy monitoring and event detection

Occupants are one of the most critical factors influencing energy consumption in the building sector. Understanding and influencing the behavior of building occupants is crucial for achieving energy conservation goals. In this section, we will focus on various energy conservation methods that prioritize the role of occupants and are based on identifying their activities, profiling their behavior, and understanding their interaction with the smart home system. These methods focus on the habit-loops of smart home occupants, which are the repetitive patterns of behavior that influence their energy consumption. By carefully analyzing these habit-loops, it is possible to detect energy wastage and make changes to reduce overall energy consumption. For example, by identifying specific choices made by occupants, such as leaving lights on or adjusting thermostats unnecessarily, it is possible to modify their behavior and lead to significant energy conservation. In the present section 1.2, our goal is to provide insights into the complex relationship between occupant behavior and building energy performance while emphasizing the importance of involving occupants as active participants in the energy management process. We will discuss three key aspects of this relationship. Firstly, we will introduce the concept of human-in-the-loop, highlighting its significance in energy management strategies. Secondly, we will present potential taxonomies to categorize the activities and behaviors of occupants. Thirdly, we will focus on the sensors used to monitor and assess occupancy and activities.

1.2.1 The Occupants - Human in the Loop (HiL)

The key factor for the smart home is integration of various sensors and controls to guarantee their appropriate operation, to optimize the energy efficiency of building and maintain the indoor comfort. However, the success of a smart house system heavily depends on the involvement of the occupants, which is not always straightforward. The

concept of Human integration in the Loop (HiL) aims to address this issue by actively involving occupants in the smart home's operation (Bavaresco et al. 2019). The inhabitants provide feedback to the system and make decisions that affect the home's energy consumption and comfort levels. Typically, a smart home system is meant to be adaptive, allowing it to modify its behavior based on the input of the residents. Manual inputs, such as altering the temperature, or automatic inputs, such as occupancy detection, can provide this feedback.

The HiL concept in smart homes offers several advantages. First, it enables the system to respond in real-time to changes in the environment and occupant preferences. This results in a more efficient system that better meets the needs of the occupants. Second, it can raise occupants' awareness of energy consumption and encourage energy-efficient practices, leading to a more sustainable and eco-friendlier lifestyle. HiL can ultimately result in higher user satisfaction, as the system is better able to fulfill the demands and preferences of the occupants.

Recent studies have focused on implementing HiL in smart homes. For example, Cho et al. 2023 present an artificial intelligence (AI) wearable sensor-based human-in-the-loop HVAC control system that is operated on a real-time basis reflecting the thermophysiological condition of the occupant to automatically improve their thermal comfort while reducing the energy consumption of the building. The wristband-type, AI-based, three-point wearable temperature sensor offers excellent thermal comfort prediction accuracy (93.9%), enabling a human-centric HVAC control operation. Wu et al. (2022) conducted a survey of existing works on HiL for machine learning from a data perspective. The authors found that HiL can be beneficial in promoting the automation of machine learning by integrating human domain knowledge into the system. The authors aimed to classify the existing works into three categories with a progressive relationship and summarize their major approaches along with their technical strengths and weaknesses. They classified existing works on HiL into three categories: (1) improving model performance from data processing, (2) improving model performance through interventional model training, and (3) designing system independent HiL. They also summarized major approaches in natural language processing, computer vision, and others. In their survey, the authors conclude that HiL can be beneficial in promoting the automation of machine learning by integrating human domain knowledge into the system even if challenges remain. One challenge is how to effectively integrate human knowledge into machine learning models, particularly in domains where data is scarce. Another challenge is determining the appro-

priate level of human involvement in the loop, as too much or too little involvement can negatively impact the performance of the machine learning system. On the other hand, opportunities for human-in-the-loop in machine learning include enabling the use of machine learning in complex or dynamic environments where traditional models may not be sufficient, as well as improving the interpretability and transparency of machine learning models (Lage et al., 2018). HiL can also help to address issues of bias and fairness in machine learning by allowing humans to provide feedback and corrections to the system. The extensive literature on the subject shows by consensus that HiL is an essential aspect of smart homes, enabling occupants to actively participate in the building’s energy management and maintain indoor comfort levels.

In this thesis, the concept of Human-in-the-Loop is considered. Given that our case studies (refer to Chapter 2) involve educational spaces, the humans in the loop, so to speak, encompass students, teachers, and even cleaning staff during certain closing hours of the facility. They are considered part of the loop because they significantly influence various parameters through their presence and activities. When present in the classrooms, students and teachers engage in various activities, including turning lights on or off, opening or closing doors and windows, using personal computers, and engaging in conversations, among others. The number of occupants and the nature of their activities may vary throughout the day. These activities can combine and overlap (refer to Chapter 5). Furthermore, they contribute to the loop once again, as their comfort requirements translate into constraints and/or objectives for optimized regulation (refer to Perspectives section).

1.2.2 The Activities – Occupants Behavior (OB)

1.2.2.1 Taxonomies of occupants’ activities

In smart home management, to maximize occupant comfort and minimize energy consumption, detection and identification of occupant activity / actions (entering/leaving the house, number of people) and house status (windows open, system failure) are critical. These activities and status information are highly unpredictable as they depend on multiple factors and are subject to instantaneous changes.

But, first it is essential to have a clear understanding of what constitutes an activity and how it can be detected and classified. Researchers have proposed various approaches for recognizing different types of activities, such as occupancy, presence, motion, location, and usage of appliances or devices. It remains difficult to establish a unified and com-

prehensive taxonomy of activities. This is partly due to the diversity of building types, occupant profiles, and cultural norms, which affect the range and variability of activities. Several studies have attempted to classify activities based on their temporal and spatial characteristics, such as their duration, frequency, intensity, and sequence (Ahn and Park, 2018). For instance, some works have distinguished between routine / primary activities, such as sleeping, eating, working, or relaxing, and non-routine / secondary activities, such as walking, talking, or listening to music, based on their context and purpose (Yamaguchi et al., 2017). Other works have used clustering or classification algorithms to group similar activities based on their sensor patterns and contextual information, such as the time of day, the day of the week, the season, or the weather (D'Oca and Hong, 2015; Dong et al., 2022). Among the many works proposing taxonomies of building occupants' activities, we can cite the classification method, known as DNAS, developed by Hong et al. (2015a; 2015b). DNAS stands for Drivers, Needs, Actions, and Systems. Its framework aims to identify the key drivers (e.g., environmental concerns, health concerns, comfort preferences, social norms, etc.) that influence occupant behavior, the needs that occupants are trying to fulfill (e.g., comfort, productivity, social interaction, personal well-being, energy conservation, sustainability, etc.), the actions that occupants take to meet those needs (e.g., adjusting temperature or lighting, opening or closing windows, using fans, engaging in social activities, using personal electronics, etc.), and the systems in place that shape those actions (e.g., building automation systems, control systems, social norms and expectations, personal habits and preferences).

Overall, the literature shows that a complete and detailed taxonomy for occupant behavior can be quite complex and depends on the specific application or domain. In Table A.3 in Appendix A, we propose a broad taxonomy that covers different aspects of occupant behavior. This taxonomy is not exhaustive and can be adapted to different applications and domains. Moreover, a taxonomy for occupant behavior can be constructed by combining activity-based, location-based, time-based, etc. This merged taxonomy can include the categories shown in Table A.4 in Appendix A.

In the following paragraphs, we will focus on the influence of occupants' behavior from various perspectives, including occupancy location detection, activities detection, occupants' habits and others (peak load shifting, energy prediction).

1.2.2.2 Occupancy location, Activity detection and identification, Habit Profiling

In the context of advanced building controls, detecting the presence and location of occupants within a building is a fundamental step. Several studies have explored methods for occupant presence detection. Das et al. (2006) have proposed the utilization of learning algorithms to pinpoint the location of multiple inhabitants, subsequently enabling the efficient management of resources based on location. This location-based resource management minimizes uncertainty and optimizes energy usage by adapting to occupants' positions. Furthermore, Barbato et al. (2009) have introduced MobiWSN, an integrated energy management system that interfaces with Wireless Sensor Networks to oversee smart home appliances. This system not only conducts user profiling through appliance sensor data analysis but also assesses the performance of prediction algorithms for occupant behaviors using simulation data. The algorithm considers various scenarios, including the Local Updating Algorithm and the Global Updating Algorithm, to ensure precise user presence and behavior predictions. These advanced techniques for occupancy location detection empower smart homes to make significant energy savings by automatically turning off redundant appliances and optimizing energy consumption patterns based on the occupants' locations and behavior. These technologies contribute not only to reducing overall energy consumption but also to enhancing user comfort and convenience in smart homes.

With the accurate detection of occupant presence established, the next vital step is the detection of their activities. Accurate detection of occupants' activities allows for the efficient adjustment of energy systems, consequently promoting energy conservation. Alhamoud et al. (2014) have introduced EnergyAdvisor, a two-step framework that first identifies user activities and then identifies appliances irrelevant to those activities. This framework facilitates energy conservation by detecting and reducing energy consumption associated with unnecessary appliances. Moreover, Cottone et al. (2015) have emphasized the importance of predicting user activities to create an energy-saving model. Monitoring user activities and scheduling appliances accordingly can help identify peak energy usage and reduce it. In a similar vein, Xu and Chen (2020) have developed a multi-step technique to identify anomalies in smart home energy usage data. This approach correlates appliance energy consumption patterns with user activities to identify normal and abnormal energy usage situations and cluster energy usage patterns. There are numerous studies focusing on Home Energy Management Systems (HEMS) that integrate with smart homes to provide energy-saving suggestions, exemplified by Abrishambaf et al. (2016) and Zhou

et al. (2014). Similarly, Machorro-Cano et al. (2020) have proposed a HEMS-IoT system that employs machine learning algorithms to offer energy-saving recommendations while considering user preferences and comfort. This system ensures smart home safety and comfort, effectively conserving energy. A case study validates the system's effectiveness in energy conservation, while users receive energy-saving recommendations through a web interface and mobile application.

Occupants' habits play a significant role in impacting energy consumption, making it imperative to address these habits effectively. The energy demand is profoundly influenced by user habits, and suggesting ways to enhance these habits can contribute to energy conservation. Researchers have explored techniques that recommend habit changes by analyzing patterns in occupancy behavior. Wang et al. (2012) have proposed a system that consists of two components: one detects people's locations in smart homes, and the other collects information on energy usage and user location. Using this information, a computer recommends ways for individuals to improve their energy usage habits. Alsalemi et al. (2019; 2020) have employed a similar approach, identifying micro-moments that represent short energy events influencing appliance states, from inactive to active, to understand energy consumption patterns based on people's activities. The authors have developed a framework that includes a micro-moment classifier, a recommendation engine that identifies people's routines, and an application that demonstrates the benefits of habit changes.

This recommend system uses sensor data to classify micro-moments and generates personalized recommendations based on individuals' energy consumption profiles, ensuring that energy-saving suggestions are delivered at the right time for increased acceptance. Furthermore, they have developed a synthetic data simulator to produce hourly energy consumption data. Himeur et al. (2020) have proposed an approach based on the identification of micro-moments to detect energy consumption anomalies using a deep neural network. Other recommend systems have been introduced by Sardianos et al. (2020) and Varlamis et al. (2022), based on the intelligent fusion of sensor data and human feedback. Additionally, analyzing patterns in appliance interaction schedules provides another avenue for reducing unnecessary energy usage, considering user actions. Diyan et al. (2020) have employed a reinforcement learning (RL) algorithm that relies on human-appliance interaction to develop an effective real-time scheduling system. The RL scheduling method segments the day into different states, with agents attached to household appliances performing various tasks to enhance their energy-saving rewards. However, it's essential to

note that the recommendations provided to the user are primarily limited to appliance interaction behavior.

1.2.2.3 Energy prediction and peak load shifting

For the energy prediction, an accurate prediction of energy load is critical for effective energy management. Jahn et al. (2010) create a user interface for monitoring appliance energy consumption, allowing off-peak scheduling to reduce bills. Similarly, the activity appliance energy consumption model proposed by Lima et al. (2015) combines activity detection and appliance usage settings to recommend energy-efficient appliance scheduling. Fakhar et al. (2023) offer a real-time off-peak scheduling technique, considering user-defined criteria. The Instant Energy Scheduling Recommendation method they have developed uses appliance energy consumption, user-created rules, and energy price signals to identify energy-saving recommendations for appliance scheduling. Chen and Lin (2018) emphasize the importance of precise energy load prediction for both homes and appliances. Arghira et al. (2012), Zhao et al. (2013) and Li and Hong (2014) present algorithms that automatically schedule appliances during low-priced hours based on energy price data, with energy storage during off-peak hours. These functions are managed by an energy management controller (EMC), which takes input from appliances, energy prices, user budget, and historical data to optimize energy distribution.

1.2.2.4 Intermediate conclusion

In the real cases examined in this thesis, which involved occupied classrooms by students, teachers, and occasionally cleaning teams, we observed a wide range of occupant behaviors that can be further contextualized using various taxonomies for occupant behavior, as we discussed previously. These behaviors are mainly governed by the original activity planned in the occupied rooms (lighting, using a projector, computers). These are also driven by objectives related to comfort preferences, energy management, and indoor environmental quality. The occupants' activities encompass a range of time-based or building control-related ones (lighting, opening/closing windows, opening/closing doors), and a significant portion of which can be characterized as unpredictable behavior. In this work, we will not focus on the problem of occupant localization, but rather on occupancy detection. Our primary focus is to discern whether a given space is currently occupied and, if so, by how many individuals. Rather than pinpointing their precise locations within occupied rooms, our emphasis lies on understanding the occupancy status itself. However,

our interests extend beyond mere occupancy. We also aim to detect and identify events that hold the potential to impact energy consumption, encompassing both electrical usage (monitored through light sensors and socket meters) and comfort considerations (such as monitoring door and window openings and closures). Furthermore, we aspire to tackle the complex challenge of detecting and disentangling multiple concurrent activities, recognizing the need for advanced approaches and techniques (as discussed in Chapter 5). Finally, an important aspect is our ability to identify behavioral patterns or habits that can greatly enhance the accuracy of estimating and predicting energy needs, demands, and usage.

1.2.3 The monitoring of occupancy and activities

After having, in the previous section, essentially presented the activities of interest that need to be detected/identified in order to characterize the influence of occupants' behavior on the energy performance of buildings, we intend in this new section to review the literature on the subject of the means mostly used for their monitoring. This new section echoes and complements subsection 1.1.2.2, primarily focused on the nature of measured data. According to the literature consulted, we can estimate that today the main studies are based on environmental sensors (section 1.2.3.1), cameras (1.2.3.2) and infrared technologies (1.2.3.3), most of which are based on wireless communication (1.2.3.4). Some alternatives have also been identified in the literature; these will also be discussed in section 1.2.3.5.

1.2.3.1 Environmental sensors for occupancy monitoring

Environmental sensors, such as CO₂, temperature, humidity and light, are commonly combined for occupancy estimation and detection (Candanedo and Feldheim, 2016).

CO₂-sensors have been found to be the most used sensor for this purpose. They are compact and non-invasive. However, the commonly used CO₂ sensors require some considerations when estimating occupancy counts. This is because human CO₂ production rates vary based on factors like current level of activities, diets, and body sizes. Also, the CO₂ concentration level will vary from one location to another based on the ventilation conditions. The CO₂-based scheme has low accuracy when estimating a large number of people and is only suitable for rough estimations. Moreover, this method has a moderate dynamic response since it takes some time for the CO₂ concentration to change when

occupants enter or leave a room. If people leave the room quickly enough that the CO₂ concentration hardly changes, their numbers may not be considered in the occupancy estimation, except if their number is important. Additionally, as the CO₂ concentration varies from room to room based on ventilation, a general model to deduce the occupancy rate using measured CO₂ levels may not be applicable. Despite these precautions and limitations to consider, the CO₂ sensor is still the most widely used for detecting the number of occupants in buildings (Gruber et al., 2014; Lapuente et al., 2022).

Multiple sensors are often combined to improve accuracy. Studies by Franco and Lecese (2020), Rahman and Han (2018), Li et al. (2019), and Kim et al. (2023) explore different methods, including Bayesian inference and machine learning algorithms, to estimate occupancy using CO₂ concentration data along with other factors. Combining various sensors compensates for limitations and enhances performance in occupancy detection. For example, Aliero et al. (2022) proposed data fusion of thermal camera video and CO₂ concentration sensors to improve the accuracy of occupancy rate predictions.

1.2.3.2 Camera used for occupancy monitoring

Occupancy estimation via cameras typically involves analyzing images or videos to detect occupants based on body features like heads, faces, body contours, or movements. Jeyapadmini and Kashwan (2015) propose a hardware-based approach using a camera for activity identification, consisting of training, comparison, and control phases. Chandran et al. (2017) use a PTZ (Pan-Tilt-Zoom) surveillance camera to detect heads for people counting. PTZ cameras monitor wide areas by dividing them into zones and capture high-resolution images for head detection. Camera-based methods offer high accuracy but face challenges like environmental obstacles, partial body exposure, and privacy concerns. Time-of-Flight (ToF) cameras overcome some of these challenges with advantages such as low-light performance, wide field of view, and the ability to detect shapes and movements without capturing identifying features (Yang et al., 2021; Navarro et al., 2022). However, ToF cameras can be costly.

1.2.3.3 Infrared technologies

The use of infrared technology, also known as infrared light, is a popular method for estimating occupancy counts due to its low cost, privacy-preserving features, and non-invasiveness. Naser et al. (2020) proposed a low-cost and low-power non-contact scheme

for occupancy estimation using an infrared thermal sensor array, which accurately extracts human body temperature from a noisy environment. Tyndall et al. (2016) used machine learning classifiers to interpret raw data obtained from a thermal detector array and a PIR sensor to determine the number of occupants in the sensor's field of view. Although infrared-based sensors offer advantages, their ability to estimate occupancy counts is questionable, and they are best suited for situations where only a small number of people need to be monitored. In addition, they cannot detect stationary occupants, and the detection range of each sensor is limited. Raykov et al. (2016) used a single PIR sensor combined with machine learning models to solve the counting problem in a room, but this method has limitations as people can easily block each other from the field of vision of a single sensor. Wahl et al. (2012) used distributed strategically placed PIR sensors which strategically placed in pairs at gateways, such as doorways and hallway sections, these sensors work in conjunction with two algorithms which are Direction Based Algorithm and a Probabilistic Distance Based Algorithm to accurately detect movement direction and estimate occupant count. This system, characterized by its cost-effective and energy-efficient design, demonstrates effectiveness in various simulated office scenarios.

1.2.3.4 Wireless communication

Wireless communication technology is commonly used to estimate occupancy counts, using technologies like Wi-Fi, Bluetooth, Bluetooth low-energy (BLE), and RFID. Wi-Fi and Bluetooth have become increasingly popular due to the ubiquity of mobile devices that support them. BLE is a more energy-efficient alternative to classic Bluetooth and Wi-Fi. Researchers have proposed various systems that utilize these technologies for occupancy detection, such as Longo et al. (2019) who compared the performance of Wi-Fi and Bluetooth/BLE for occupancy estimation. Lu et al. (2016) proposed a system that uses commercial Wi-Fi hardware for occupancy inference, while Zou et al. (2018b) developed a Wi-Fi-based device-free method for occupancy detection and crowd counting. However, Wi-Fi-based systems have limitations, such as the possibility of multiple smartphones per occupant, occupants not turning on Wi-Fi on their devices, or occupants forgetting their phones. And alternatives, such as BLE based systems also require additional cost and maintenance for deployment (Iannizzotto et al., 2023).

1.2.3.5 Alternative monitoring

In addition to common sensors such as CO₂, temperature, humidity, cameras, and PIR, there are also unconventional techniques for occupancy detection in buildings. Light sensors (photodiodes) can detect occupancy by measuring changes in light levels caused by the movement, but they may not work well in areas with ample natural light. Machine learning with light sensors can improve accuracy (Candanedo and Feldheim, 2016). Another technique investigated in literature is audio extraction, which involves analyzing sound waves to detect human presence when privacy concerns limit camera use. Huang et al. (2018) propose an audio-based occupancy estimation technique. Ultrasound sensors emit sound waves, bounce them off objects, and analyze reflections to detect humans, often used with other sensors for accuracy (Hammoud et al., 2017). Microwave Doppler radar, as used by Islam et al. (2023), is unobtrusive and privacy-friendly for occupancy estimation. Depth sensors like the Microsoft Kinect were used for crowded area counting while preserving privacy (Diraco et al., 2015). While these non-conventional techniques offer some advantages over traditional methods, they also have some limitations, such as their reliance on ideal conditions for accurate detection and their susceptibility to detect occupants when a room is empty or reversely to detect an empty room whereas people are present. Chair sensors are another unconventional technique that detect human presence by analyzing pressure and weight distribution when someone sits on a chair (Labeodan et al., 2016).

Each sensor type has advantages and limitations, summarized in Table A.5 (in Appendix A). Combining data from multiple sensors is key to accurate occupancy estimation. However, it is also important to note that even with the most advanced sensor technology, occupancy prediction is only useful if there are effective methods of analyzing and modeling the data. This is where the field of data analytics comes into play. Machine learning algorithms can analyze sensor data to develop sophisticated occupancy models (see section 1.3).

After careful consideration, we have ultimately chosen eight sensor types (CO₂ sensor, temperature sensor, humidity sensor, Light sensors, electricity power consumption sensor, door open sensor, window open sensor, volatile organic compound sensor) from those presented in the table A.5 in Appendix A. Our objective was to encompass a broad range of possible measurements, all of which provide valuable information for occupancy estimation, occupant comfort, energy consumption, and indoor air quality. While precision was a crucial criterion in our selection process, it was not the sole determining factor. We also

aimed to utilize the maximum number of commonly used sensors with well-established functionality and maintenance protocols. The use of PIR sensors was considered due to their capability for entry and exit counting. However, we determined that these sensors were more susceptible to "student interferences" and potential long-term reliability issues. Additionally, certain ranges of PIR sensors are known to be sensitive to temperature and humidity variations, limiting their ability to detect or characterize the activities of individuals within the room, whether they are stationary or in motion. With these considerations in mind, we have assembled a suite of selected sensors that collectively address our research objectives while minimizing potential drawbacks. These sensors have demonstrated robust performance, are easily accessible for maintenance, and offer the necessary precision and reliability for our occupancy estimation and energy management applications.

1.3 Detection, identification and estimation

In the previous section 1.2, we have presented some taxonomies to categorize the activities and behaviors of occupants and have focused on the sensors used to monitor and estimate occupancy. The aim of the present section 1.3 is to focus on the detection, identification and estimation of occupants' activities using sensors and data analytics. It is essential to have a clear understanding of what constitutes an activity and how it can be detected and classified. In this section and the following paragraphs, any activity of the occupants and any change in the state of the house will be seen, at first view, as a deviation from the "normal" operation of the building. The "normal" function is limited to the response of the building to climatic, material variables, thus excluding any variable associated with the above-mentioned occupant activities and change of state. In this first vision, the activity detection problem can be considered as an anomaly detection one. The problem of identifying one or more types of activities or changes of state will be a classification problem (section 1.3.2). Anomaly detection is also known as deviation detection because the value of an anomaly object always deviates significantly from the expected or common data value. This deviation can be quantitatively estimated using regression methods (section 1.3.3). However, before moving on to a review of classification and regression methods, Section 1.3.1 will provide the important elements for understanding the differences, characteristics, and trade-offs between so-called supervised, unsupervised and semi-supervised learning methods, and ultimately for choosing the appropriate approach for a given problem. At the end of section 1.3, we will propose, regarding the informa-

tion found and compared in the literature, an overall table presenting a synthesis of the advantages and disadvantages of the methods presented in the two subsection 1.3.2 and 1.3.3.

1.3.1 Main categories of learning methods

In the field of occupancy’s detection / estimation, three main categories of learning methods are widely used: supervised learning, unsupervised learning, and semi-supervised learning. Each of these approaches has its own unique characteristics, applications, limitations, and constraints. Supervised learning is a learning paradigm where the model is trained on labeled data, meaning that the input samples are accompanied by their corresponding target values or labels. The model learns to map the input data to the desired output based on the provided examples. Some popular supervised learning algorithms include Linear Regression, Support Vector Machines (SVM), Decision Trees, Random Forests, and Neural Networks. Supervised learning excels when abundant labeled data is available, but it may face challenges in scenarios with limited labeled data or when the labeling process is expensive.

Unsupervised learning, on the other hand, deals with unlabeled data, where the algorithm learns to identify patterns, structures, or relationships in the data without any prior knowledge of the desired outputs. Clustering algorithms such as K-means, DBSCAN, and Hierarchical Clustering are widely used in unsupervised learning to group similar data points together. Dimensionality reduction techniques like Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE) are employed to capture the underlying structure and reduce the dimensionality of the data. Unsupervised learning finds applications in areas such as anomaly detection, data exploration, and recommendation systems. One of the main challenges in unsupervised learning is evaluating the quality and interpretability of the learned representations.

Semi-supervised learning lies between supervised and unsupervised learnings, leveraging a combination of labeled and unlabeled data. It aims to improve the performance of models by utilizing the limited labeled data along with the abundance of unlabeled data. The labeled data provides crucial supervision signals, while the unlabeled data helps in capturing the underlying distribution and expanding the training set. Techniques like Self-Training, Co-Training, and Generative Adversarial Networks (GANs) are commonly employed in semi-supervised learning. This approach finds applications in scenarios where acquiring labeled data is costly or time-consuming. However, it still faces challenges in

scenarios with imbalanced (i.e., with very few abnormal data compared to the amount of normal data) labeled data and ensuring the quality and reliability of the inferred labels from the unlabeled data.

It is important to note that these categories are not mutually exclusive, and hybrid approaches often exist. For example, transfer learning combines elements of supervised and unsupervised learning to leverage knowledge from one domain to another. Reinforcement learning is another learning paradigm where agents learn to interact with an environment and make decisions based on rewards and punishments. The choice of a method depends on factors such as the availability of labeled data, the nature of the problem, the desired interpretability of results, and the scalability of the algorithms.

1.3.2 Detection and identification of activities – a classification problem

As previously stated, activity detection and identification can be considered as an anomaly detection problem and falls under the category of classification problems. In view of the importance of the machine learning (ML)-based approaches, we will divide subsection 1.3.2 into three parts. In the first one (§ 1.3.2.1), we quickly give the definitions of some metrics used to assess the quality of the classification work of algorithms. The second part (§ 1.3.2.2) will deal with traditional (i.e., non-network based) ML classification / fault detection methods. The third part (§ 1.3.2.3) will focus on those learning methods based on the use of networks or on deep learning (DL)-based approaches. The methods used for the detection of building occupancy can be classified into different categories depending on the purpose of the data collection or on the data analysis approach employed. Thus, one way to categorize these methods is by purpose: detection of presence, identification of occupants' activities, or estimation of the number of occupants. This latter type - estimation of the number of occupants - can be seen as a classification problem (each number of occupants corresponds to a specific class) as well as a regression problem (estimation of the number by a model). The main algorithms used for the estimation of the number of occupants will be also reported in the present section. Another way is to differentiate between supervised, unsupervised, and semi-supervised classification methods. Both categorizations are useful for understanding the state-of-the-art methods used for occupancy detection.

Table A.6 (in Annex A) complements this state-of-the-art by summarizing and report-

ing the methods used, their accuracies, as well as the types of data and sensors used.

1.3.2.1 Some definitions of metrics used in classification

Some (basic) concepts and indicators are used to monitor the efficiency and quality of the classification. Most of the common indicators we limit ourselves to are derived from the confusion matrix. The confusion matrix is a table used to evaluate the performance of a classification algorithm (see Table 1.1 below for an example on presence detection). The confusion matrix contains the number of true positive, true negative, false positive, and false negative predictions made by the algorithm.

		Predicted	
		Absence	Presence
Real	Absence	True negative (TN)	False positive (FP)
	Presence	False negative (FN)	True positive (TP)

Table 1.1 – Confusion Matrix

From this confusion matrix can be calculated or drawn:

- Accuracy: A measure of the overall performance of a classification algorithm and gives the proportion of correctly classified instances ($Accuracy = \frac{TP+TN}{TP+TN+FN+FP}$).
- Precision: A measure of the ability of a classification algorithm to avoid false positive predictions and give the proportion of true positives among all predicted positives ($Precision = \frac{TP}{TP+FP}$).
- Recall (or Sensitivity): A measure of the ability of a classification algorithm to find all the positive instances in the data and gives the proportion of true positives among all actual positives ($Recall = \frac{TP}{TP+FN}$).
- Specificity: A measure of the ability of the model to detect negative instances correctly and give the proportion of true negatives among all actual negatives ($Specificity = \frac{TN}{TN+FP}$).
- F1-score: A measure of the performance of a classification algorithm that balances the recall and precision and corresponds to the harmonic mean of precision and recall ($F1 = \frac{2 \times (\text{Recall} \times \text{precision})}{\text{Recall} + \text{Precision}}$).
- ROC curve: Draws a plot of true positive rate vs false positive rate and is used to evaluate the trade-off between recall and specificity.

1.3.2.2 Traditional classification methods

There are many examples of unsupervised clustering methods being used for occupancy detection in buildings. One example is the work of Pan et al. (2017) who present a case study in Shanghai residences using K-means cluster analysis for occupant-behavior-based electricity load patterns in buildings. The study analyzes the electricity consumption patterns of residents through smart meters and proposes an approach to group similar consumption patterns based on their features. The study also identifies the characteristics of each cluster and provides recommendations for energy management strategies. Unsupervised K-means data mining approach has also been used by Yang et al. (2023) who aims identifying and analyzing the energy consumption patterns and characteristics of college dormitories in detail and examining three influencing factors (occupants' gender and floor and orientation location of rooms). A study on the use of occupancy and energy data to develop a predictive model for building energy consumption in commercial buildings is presented by Yang et al. (2017a). The authors propose a novel method of clustering to identify patterns in occupancy and energy usage data and use regression models to predict future energy consumption based on these patterns. The results show that the proposed method can accurately predict energy consumption in commercial buildings and has the potential to be used for energy-efficient building operations.

Among the best-known and most used supervised clustering methods in the field of occupancy estimation, the k-Nearest Neighbors (k-NN), Support Vector Machine (SVM) and Decision Trees methods should be mentioned. The underlying idea of the k-NN method is quite intuitive: if the majority of the nearest neighbors of a sample in the feature space belong to a certain category, then the sample should also belong to that category. Nonetheless, Szczurek et al. (2017) demonstrated that k-NN, as a nonparametric, nonlinear, minimum distance classifier, is effective for occupancy determination. In a study by Vela et al. (2020), k-NN was found to be the best algorithm for occupancy estimation, but its use may be limited due to its $O(n^2)$ time complexity and its high computational cost. Another issue is that k-NN may not perform well when the sample size is imbalanced, i.e., when one class has a much larger sample size than others, which can cause confusion among the k-neighbors of a new sample. The SVM method is a type of classifier that is used for supervised learning in data classification and regression. It uses a decision boundary known as the maximum margin hyperplane, which is based on the samples used for learning. In the literature reviewed, SVM is one of the most commonly used machine learning algorithms. Zuraimi et al. (2017) used the SVM algorithm combines

with artificial neural networks (ANN) to identify the number of people in a space using CO₂ sensors. Chandran et al. (2017) used a cascade detector of SVM to detect human heads. Shih (2014) used a modified SVM-based observational measurement to provide robust day-and-night occupant tracking and counting performance. Literature globally agrees on the fact that SVM methods work well with high-dimensional data and is very fast in prediction when the number of classes is limited. Involving a decision tree model, D’Oca and Hong (2015) tried to identify occupancy patterns and translate them into typical working user profile schedules that can be used as input to building energy modeling programs. The process involves a rule induction algorithm to learn a pruned set of rules, and a cluster analysis to obtain consistent patterns of occupancy schedules.

Two other more recent types of classification methods with great potential have recently been exploited for fault detection and diagnosis (FDD) and might be good candidates for occupancy detection and identification. The first type of methods is the one based on the density-based spatial clustering of applications with noise (DBSCAN) algorithms, relying on the identification of clusters as areas of high data point density separated by areas of lower density (see further details in section 4.2.1). Yan et al. (2016) utilized the density-based clustering algorithm OPTICS (Ordering Points to Identify The Clustering Structure) to identify sensor faults in Air Handling Units (AHUs). Novikova et al. (2020) also employed a density-based clustering algorithm to detect anomalies in Building Automation Systems (BAS) data of a three-story building. These works demonstrated that cluster analysis is a promising method to condense a large volume of BAS data into a few operational patterns that can be suitable for HiL identification and building systems faults interpretation. The second promising type of methods is the Spectral Clustering (SC) technique, leveraging the eigenvalues and eigenvectors of a similarity matrix derived from the data and which is effective in handling non-convex and complex-shaped cluster. In the field of building monitoring, SC technique has been used by Ghaffar et al. (2022) for extracting individual appliance energy usage from the aggregate energy profile of the building. These two techniques, DBSCAN-based and SC-based, are competitive and viable, with advantages of low complexity, high accuracy, no training data requirement, and fast processing time.

Methods borrowed by machine learning from statistics have been also applied to occupancy and event detection. The most frequently applied methods are the logistic regression (LR), Gaussian mixture models (GMMs) or processes (GMPs), random forests (RFo), and Naïve Bayes Classification (NBC). LR is a supervised classification method that seeks to

predict the probability of a sample belonging to a particular class. It is commonly used in binary classification tasks, where there are only two possible classes. For instance, Stazi et al. (2017) applied and compared a logistic regression model and a linear regression model to detect the windows status (opened or closed) in classrooms. GMMs are a probabilistic clustering technique that assumes the data points are generated from a mixture of Gaussian distributions. In the context of occupancy estimation, GMMs can be used to cluster data from various sensors in order to identify occupancy patterns in a building. GMMs can model complex occupancy patterns that arise due to the presence of multiple occupants, their different schedules and activities, and the effect of external factors such as weather. One advantage of GMMs is that they can provide uncertainty estimates for occupancy predictions, which can be useful in decision-making processes (Xu et al., 2020). RFo is a supervised machine learning algorithm that builds multiple decision trees, mentioned above, to generate a prediction model. RFo can be used for classification or regression tasks and is particularly useful when dealing with high-dimensional data (Parzinger et al., 2022). NBC is a probabilistic algorithm based on Bayes' theorem, which assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. In the context of occupancy detection, NBC can be used to classify occupancy patterns based on the probability of different events occurring together (Fajilla et al., 2021; Aliero et al., 2022).

1.3.2.3 Advanced classification methods based on networks

Before exploring network-based classification methods, let us introduce the Principle Component Analysis (PCA) method. This method, used in fault detection and diagnosis, shares key concepts with neural networks: reducing variables in a low-dimensional latent space, constructing models in that space, and reconstructing data. By comparing the original data to model-calculated data, we estimate a reconstruction error for each data point. If this error is too large, it indicates differences from the majority and signifies an abnormal situation. In the field of occupancy estimation, which is of interest to us, we can mention for example the study by Baird et al. (2017).

However, in the evolving field of machine learning, advanced techniques like Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2020) have emerged as potential alternatives to PCA for dimensionality reduction. UMAP, unlike PCA's linear approach, utilizes non-linear relationships and local structure preservation, excelling at capturing complex patterns and nonlinear data relationships. This makes it suitable for

tasks like occupancy estimation. UMAP is a rather novel learning technique for dimension reduction and the literature in a field similar to ours is still very rare. In a similar context, Khan et al. (2023) recently applied dimensionality reduction techniques to city-scale data to identify key features of high consumption and generation areas based on building character from 72,000 households in the Netherlands. UMAP competes favorably with t-SNE in terms of visualization quality and preserving complex, non-linear relationships, offering superior runtime performance.

Turning now to network-based machine learning approaches, the states of the art conducted by Dai et al. (2020) and Zhang et al. (2022a) on fault detection-based methods for occupants' activity or presence detection show that the most recently used methods are machine learning methods from shallow neural networks¹(SNN) to deep learning methods such as artificial (ANN), recurrent (RNN), convolutional neural networks (CNN).

SNNs, situated within the ANN family, consist of just one hidden layer between input and output layers. They are faster to train on smaller datasets due to their simpler structure but may have performance limitations in complex occupancy detection scenarios, as noted by Chalapathy et al. (2021) in their study on building cooling load prediction using shallow and deep learning models.

ANNs, a class of machine learning algorithms, consist of interconnected nodes known as artificial neurons or perceptrons, organized in layers. They excel at recognizing patterns and relationships between inputs and outputs. ANNs can address both classification and regression problems, depending on the activation function in the output layer. The sigmoid function enables binary or multiclass classification, while a linear or identity function suits regression tasks. Although ANNs support both supervised and unsupervised learning, they are predominantly used for supervised learning.

In the context of smart homes, ANNs find extensive application in occupancy detection and energy conservation (Zhang et al., 2022b). These algorithms leverage data from various sources like occupancy sensors, smart meters, and environmental sensors

1. A neural network is a powerful machine learning algorithm that mimics the structure and functioning of the human brain. It is composed of interconnected layers of artificial neurons, which receive inputs, perform computations, and produce outputs. Through a process called training, neural networks can learn patterns and relationships in complex data, enabling them to make predictions, classify objects, and solve a wide range of problems. With their ability to handle nonlinear relationships and extract high-level features, neural networks have achieved remarkable success in various fields, including computer vision, natural language processing, and speech recognition. By adjusting the network architecture and optimizing its parameters, neural networks can adapt to different types of data and improve their performance over time. Their capacity to handle large datasets and model intricate patterns makes them a valuable tool in modern data-driven applications.

to monitor occupants' activities and habits. They make real-time decisions on appliance control and energy optimization. Home energy management systems (HEMS) have also been integrated with smart homes to offer energy-saving suggestions (Abrishambaf et al., 2016; Zhou et al., 2014), employing machine learning to understand user behavior and energy consumption patterns (Machorro-Cano et al., 2020). For instance, one system clusters homes based on energy loads and delivers energy-saving advice using RuleML² and Apache Mahout³ while ensuring smart home safety and comfort.

In the field of activity recognition, Fang and Hu (2014) employed motion sensors to record human activities and found that the Back Propagation (BP) neural network outperformed other probabilistic algorithms, like the Naïve Bayes (NB) classifier and Hidden Markov Model (HMM) in terms of recognition accuracy. Oniga and Sütő (2014) utilized multiple ANNs to analyze signals from acceleration sensors for daily life activity detection. Paradiso et al. (2013) employed a multilayer back-propagation neural network to detect appliance power usage across eight monitored devices, achieving an accuracy of 95.26%. For real-time occupancy detection, Kampezidou et al. (2021) developed a physics-informed pattern-recognition machine (PI-PRM) that extracts invariant features such as CO₂ level, CO₂ rate of change, and HVAC states. The proposed PI-PRM method provides real-time estimation, achieving 97% accuracy within minutes of an occupancy change. Similarly, Danaei-Mehr et al. (2016) employed three distinct ANN methods for detailed human activity recognition. The Levenberg Marquardt (LM) algorithm yielded the highest detection rate at 92.81%.

Despite the achievements of ANNs, they come with certain limitations, including extended training times, sensitivity to initial weight values, and challenges in handling extensive datasets. Addressing these issues, the eXtreme Learning Machine (XLM) offers faster training, reduced parameter tuning, and enhanced generalization performance. XLM, a type of Artificial Neural Network initially proposed by Huang et al. (2006), deploys a single hidden layer feedforward network structure. Unlike traditional ANNs, XLM uses randomly generated and fixed weights between input and hidden layer neurons. The output layer weights are computed using a single linear regression step. Consequently, XLM's

2. RuleML is a knowledge representation language used to express rules and logic in a machine-readable format. It enables the formal representation of rules and inference capabilities, making it suitable for applications such as rule-based reasoning and expert systems.

3. Apache Mahout is an open-source machine learning library that offers a diverse set of algorithms and tools for building intelligent applications. It is designed to handle large-scale datasets and can be integrated with other Apache frameworks for distributed processing. The library provides functionalities for tasks like clustering, classification, recommendation systems, and collaborative filtering.

training process is much quicker than traditional ANN methods, as it does not involve iterative weight updates through backpropagation or other techniques.

Chen et al. (2016) investigated occupancy presence detection using indoor environment sensors and various algorithms, including XLM, ANN, SVM, k-NN, linear discriminant analysis (LDA), classification and regression tree (CART), gradient boosting machine (GBM), and RFo methods (the last three methods are more specifically categorized as ensemble learning or EL methods, as mentioned in paragraph 1.3.2). They found that XLM outperformed the others in terms of accuracy (99.3%), closely followed by SVM and RFo. XLM also boasted the advantage of being significantly faster than the other methods, rendering it more practical for real-time applications. However, the authors noted that XLM's effectiveness might diminish when handling high-dimensional data, and the choice of input features could significantly influence the performance of all methods.

While ANNs or XLMs have shown promise in smart home applications like occupancy detection and energy conservation, they have limitations. For instance, they struggle to model sequential data, which is common in smart homes, including occupancy detection and energy forecasting. Recurrent Neural Networks (RNNs) are deep learning algorithms capable of addressing this limitation. RNNs maintain an internal state or memory to consider previous inputs when processing current ones, making them suitable for tasks involving time-series data, natural language, and speech.

In our area of interest, RNNs have been applied to various building energy systems applications, such as occupancy detection, energy consumption prediction, and fault detection. Xu and Chen (2020) developed an unsupervised anomaly detection framework using deep learning, including RNN and quantile regression. This framework can predict abnormal energy consumption and classify anomalies into severity grades based on quantile ranges. RNNs offer advantages like handling variable-length sequential data, capturing long-term dependencies, and adapting to various input types. Long Short-Term Memory (LSTM) networks within RNNs are effective for modeling complex temporal relationships. While RNNs and LSTMs have been primary choices for occupancy detection, they face limitations in scenarios involving multiple input and output signals. To overcome this, RNN-MIMO (Multiple-Input-Multiple-Output) models have been introduced, allowing simultaneous processing of multiple input sensors and output signals, leading to improved occupancy detection accuracy. Chalapathy et al. (2021) achieved outstanding performance with the RNN-MIMO model, reaching 98.8% accuracy and an F1-score of 0.99. Multiple sensors enhanced prediction accuracy and reduced false alarms. Another algorithm,

M-FRNN (Markov-based Feedback RNN), is useful for predicting time series with missing data. Wang et al. (2018a) applied M-FRNN to estimate occupants number in an open office space using Wi-Fi probe data, environmental measurements, and camera data as ground truth. M-FRNN demonstrated around 80% accuracy within a two-occupant tolerance for 60 occupants. However, RNN-MIMO and M-FRNN may require more computational resources compared to traditional RNNs and may not be ideal for real-time applications.

As discussed earlier, ANNs and RNNs have their merits, but they have limitations when dealing with tasks that involve extensive image data analysis, such as image classification, object detection, and segmentation. Convolutional Neural Networks (CNNs) excel in these scenarios. CNNs employ convolutional layers to apply filters to input data, followed by pooling layers to downsample the output, enabling automatic feature extraction. This makes them highly effective for tasks like autonomous driving, facial recognition, and medical image analysis, thanks to their ability to detect subtle features. Furthermore, CNNs have fewer parameters than ANNs, reducing the risk of overfitting and enhancing generalization to new data.

In the domain of occupancy prediction, many people-recognition methods based on CNNs have been developed, delivering high accuracy and precision. Conti et al. (2014) introduced two CNN-based algorithms for counting people in a classroom, both achieving excellent results. Bao et al. (2021) devised a people-counting algorithm utilizing low radiation impulse radio ultra-wide bandwidth radar, which employs CNN for enhanced stability in scenarios with obstructions and superposition. Researchers have also applied CNN to crowd counting in complex scenes, demonstrating success. Recent studies tackle population counting by regressing a population density image, summing the density values to determine the number of people in an image. This approach handles severe occlusion in crowded images effectively.

Also relying on CNN, Lee et al. (2017) used various data sources, including indoor environmental data and triaxial accelerometer data from users' smartphones. They introduced a one-dimensional CNN method to discern specific occupant activities, like walking, running, and sitting. This method achieved an accuracy of 92.71%. In a more recent development, Tang et al. (2020) introduced a Passive Wi-Fi Radar (PWR) technique for occupancy detection and people counting. PWR operates within any environment covered by an existing WiFi local area network without the need for special modifications to the Wi-Fi access point. The proposed PWR system boasts a remarkable 99.5% accuracy in

determining room occupancy and accurately counting people (98.1%) when up to four people are present. This study sheds light on the potential applications of PWR techniques in human resources management, optimizing energy usage in smart buildings, and enhancing public services in future smart cities.

While previous deep learning methods have achieved success in occupancy prediction, they often demand substantial labeled data or manual feature engineering, which can be time-consuming and costly. To address these constraints, researchers have turned to learning techniques like Auto-Encoders (AEs). AEs can autonomously learn valuable representations of input data and extract low-dimensional representations from high-dimensional data. An interesting aspect is their ability to generate new data by sampling from the learned compressed representation and reconstructing it into the original data space. AEs are sometimes classified as semi-supervised methods since they can utilize both labeled and unlabeled data during training. The unsupervised aspect arises from AEs reconstructing input data without explicit supervision, while the supervised facet comes into play when fine-tuning AEs on labeled data, enabling tasks like classification or regression. By leveraging both labeled and unlabeled data, AEs can potentially enhance the performance of these supervised tasks. For instance, using an AE as pretraining for a deep neural network can mitigate overfitting and enhance the model's generalization.

Other semi-supervised methods exist but are relatively rare in the field of occupancy prediction. These include manifold regularization⁴, mainly used in fault diagnosis techniques [example of AHUs given in Yan et al. (2018)], algorithms based on transfer learning principles, such as domain adaptation⁵ techniques [example of Human Occupancy Counting method using CO₂ sensor data by Arief-Ang et al. (2017)] or the transductive SVM⁶ and the label propagation⁷. These methods hold promise for addressing unbal-

4. Manifold regularization: This method assumes that data points that are close in the input space should have similar outputs. It uses a regularization term that encourages the classifier to produce similar outputs for similar inputs.

5. Domain Adaptation: Semi-supervised domain adaptation is an approach to transfer learning (transfer of knowledge gained in one task to another domain or task) in which labelled and unlabelled data from a source domain are used to improve classification performance in a different but similar target domain.

6. Transductive support vector machines (TSVM): This method uses the labeled data to learn a boundary in the input space and then applies this boundary to the unlabeled data to classify it.

7. Label propagation: This method assumes that data points that are close in the input space should have similar labels. It uses the labeled data to learn a labeling function and then propagates these labels to the imbalanced data based on their proximity in the input space.

anced, sparse, and scarce databases but have not yet found widespread application in our field.

Similar to traditional classification methods not based on networks, machine learning has borrowed techniques from statistics to apply them to occupancy and event detection. Among the approaches mentioned, Bayesian networks (BNs) and their dynamic extension (DBNs) are widely used. BNs, also known as belief networks or graphical models, are a type of probabilistic graphical. These models represent the joint probability distribution of random variables using a directed acyclic graph (DAG), where nodes represent variables and edges signify probabilistic dependencies. It is important to note that BNs differ from deep learning methods like ANNs, CNNs, and RNNs. BNs are probabilistic models explicitly handling uncertainty and causal relationships, while deep learning methods use deterministic functions to learn patterns. BNs excel in cases of limited or noisy data and provide a probabilistic representation of uncertainty. They are also more interpretable than deep learning methods, as they explicitly model variable relationships. Furthermore, BNs can extend to dynamic systems through DBNs (Bigaud et al., 2019) and hidden Markov models (HMMs), making them useful in fields with sequential and dynamic data.

Additionally, BNs can be used in both supervised and unsupervised tasks. In supervised tasks, they are used for classification and prediction using input data and corresponding class labels. In unsupervised tasks, they can be used for learning the dependency structure between variables in the data without corresponding class labels. BNs have diverse applications, including energy-efficient system design (Tian et al., 2019), occupant comfort assessment (Bortolini and Forcada, 2019 ; Hosamo et al., 2023), and technical equipment fault detection (Bigaud et al., 2019). In building activity detection, BNs model relationships between sensor data, (e.g., temperature, light, occupancy), human activities (e.g., cooking, watching TV, sleeping), and technical events (e.g., lighting, HVAC, or office equipment usage). For instance, Amayri et al. (2019) employed BNs to detect activities (such as cooking, watching TV, and sleeping) in a smart home, outperforming other algorithms. He et al. (2014) used BNs to accurately detect activities in a multi-occupancy office building. Tian et al. (2019) used BNs to identify activities impacting energy consumption in a commercial office building. They collected sensor data encompassing temperature, light, motion, and power usage, which was then employed to train a Bayesian network (BN) capable of identifying activities like lighting, HVAC, and office equipment utilization. In contrast, Amayri et al. (2019) proposed an unsupervised method for characterizing occupant behavior (occupancy and activity) in both office and residential buildings. This

approach utilizes domain knowledge obtained from questionnaires and recorded sensor data for motion detection, power consumption, hot water usage, and indoor CO₂ levels.

Despite the advantages of BNs, deep learning methods have shown superior performance in numerous tasks, especially in domains with large amounts of high-dimensional data. However, in scenarios where interpretability, uncertainty modeling, and sequential data modeling are important, BN and their extensions may offer a valuable alternative or complementary approach.

To conclude this sub-section 1.3.2, it can be noted that ML methods have achieved impressive performances but face several challenges like overfitting, vanishing gradients, and limited data availability. To address these, researchers are exploring hybrid methods that combine multiple machine learning architectures. This approach of hybridizing Deep Learning (DL) methods and comparing possible combinations is part of the overall learning approach that combines multiple models to improve the accuracy and robustness of classification. An example of hybrid method is the Auto-encoder Long-term Recurrent Convolutional Network (ALRCN), developed by Zou et al. (2018c), which combines AEs, RNNs, and CNNs. ALRCN is specifically designed for spatiotemporal data, such as time series or videos, and it can learn both spatial and temporal features simultaneously. The AE component reduces the dimensionality of the input data and captures its main features. The CNN component extracts spatial features from the encoded data, while the RNN component models the temporal dynamics of the data. By combining these three architectures, ALRCN can achieve state-of-the-art performance on a wide range of spatiotemporal prediction tasks. This way of hybridizing methods can be roughly likened to a series assembly of methods. We adopt this approach in the thesis chapter dedicated to multiple activity detection, where ANN, PCA, Hotspot (see Chapter 5), and BN methods will be used in cascade (the justification for the hybridization of such methods will be provided in the section dedicated to its presentation). With the same ulterior motive of combining several algorithms to improve the prediction and the robustness of the results, researchers have evolved towards the so-called ensemble learning (EL) approaches. Drawing upon the same analogy as in the preceding paragraph, this alternative approach to method hybridization resembles more of a parallel ensemble, where each method utilizes the data, whether shared or not, collectively and simultaneously to achieve the same objectives. The basic idea behind EL technique is that by aggregating the predictions of multiple models, the resulting ensemble can often outperform any single model. The ensemble is typically created by training multiple models on the same dataset using dif-

ferent algorithms, different subsets of the data, or different feature representations. The individual models can be of the same type (homogeneous ensemble) or different types (heterogeneous ensemble).

EL can be applied to various machine learning algorithms, including ANN, RNN, CNN, AE, etc. Two primary EL approaches are Bagging and Boosting. In Bagging, each ensemble model is independently trained on a randomly sampled subset of the training data (with replacement). The final prediction is then obtained by aggregating the models' predictions, often through majority voting for classification problems (or averaging for regression problems). For instance, Wang et al. (2018b) developed an Ensemble Bagging Tree model (EBT) to efficiently predict short-term electricity demand in institutional buildings, achieving reduced computation time without sacrificing prediction accuracy. Boosting is an iterative EL technique where models are trained sequentially, with each subsequent model focusing on instances misclassified by the previous ones. The final prediction combines all models' predictions, weighted by their individual performance. For instance, Cao et al. (2020) compared single models and EL algorithms for predicting daily electrical load in healthcare buildings, demonstrating that Extreme Gradient Boosting (XGBoost), a popular boosting algorithm, outperforms single models. We employ a similar boosting approach in our research using XGBoost techniques (see Chapter 4).

All the candidate ML algorithm are summarized and compared in Table A7 (in Appendix A).

1.3.3 Estimation and prediction – a regression problem

As previously stated, activity detection and identification can be classified as an anomaly detection problem and fall under the category of classification problems. In contrast, prediction requires leveraging the potential for regression or meta-model construction offered by learning methods. Building a regression model aims to estimate the probability and performance level of an event. Regression models are particularly useful for making predictions over time, incorporating dynamic functions. Estimation and prediction provide valuable insight that can be further utilized in control and regulation models. These models focus on detecting or anticipating performance drifts, correcting them, or optimizing the function of the complex system under study.

Many of the methods discussed earlier for detection and identification can be extended to regression tasks. The key distinction lies in the nature of the output variable. In classification, the output variable is categorical, assuming a finite number of distinct values,

often representing different classes. The objective is to assign each observation to the appropriate category based on its input features. In regression, the output variable is continuous, allowing it to assume any numerical value within a specific range. Here, the goal is to predict the numerical value of the output variable based on the input features.

1.3.3.1 Traditional regression methods

Before considering machine learning methods themselves, it is necessary to discuss the pioneering time-series methods that have been extensively studied. In 2010, Newsham and Birt used the Autoregressive Integrated Moving Average (ARIMA) method to enhance building electricity use forecasts. They demonstrated that incorporating occupancy data can enhance accuracy, and they employed ARIMA models to account for occupancy effects. Although these methods have been improved over time, including variants and improved versions like Seasonal Autoregressive Integrated Moving Average (SARIMA) and Seasonal Decomposition of Time Series (SDTL). Chen and Soh (2017) and Wang et al. (2019b) have indicated that machine learning-based approaches generally outperform these methods. Recently, Facebook introduced a promising and efficient method called "Prophet," which uses an additive model with seasonality and trend components for time-series forecasting, particularly effective for data with strong seasonal patterns (Parise et al.2021). Despite these improvements, the prevailing trend among researchers is to hybridize these temporal (or even frequency) decomposition approaches with machine learning techniques to improve prediction. For instance, Yuan et al. (2021) presented a hybrid approach integrating temporal-sequential analysis and machine learning for building occupancy prediction, showing potential for improving building energy efficiency.

Returning to the machine learning methods that we have already mentioned for their ability to do the classification work, and following time series processing, we can cite statistical methods such as logistic or linear regressions. Linear regression predicts the target value as a linear combination of input variables, making it easy to model and widely used in statistical analysis. Kim and Srebric (2017) used linear regression to explore the correlation between occupancy and electricity consumption. In contrast, logistic regression models the relationship between features and the target using a logistic function, predicting the probability of a new test sample belonging to a certain category. Yuan et al. (2019) applied the Softmax Regression Model, a generalization of logistic regression for multi-classification, to elucidate the dynamic relationship between environmental parameters and indoor occupancy. These simple machine learning methods, based on linear

or logistic regressions, clearly compete with the Prophet or hybrid methods mentioned earlier.

When the relationship between the input variables and the output variable is nonlinear, and data contains outliers, Support Vector Regression (SVR) may outperform logistic or linear regressions. Unlike these, SVR is a nonparametric method that does not assume data distribution. It also incorporates a regularization parameter that prevents overfitting and enhances model generalization. Moradzadeh et al. (2020) assessed two machine learning techniques, namely ANN and SVR, for predicting residential building heating and cooling loads. The study demonstrated that both methods yield accurate predictions, but SVR surpasses ANN in terms of accuracy and computational efficiency. Moreover, the results suggest that combining weather data with building data significantly enhances load forecasting accuracy.

Always considering traditional methods used for regression/prediction of performance and/or occupancy, Candanedo et al. (2016) assessed Hidden Markov Models (HMM) accuracy in estimating occupancy using various environmental parameters (temperature, humidity, humidity ratio, CO₂, and light time series data). The accuracy varied based on data collection and fusion of these parameters, emphasizing the effectiveness of sensor fusion. HMMs offer the ability to model intricate variable dependencies and capture multiple sources of variability simultaneously compared to time-series methods. However, they may be less interpretable and demand substantial data to train effectively. HMMs are often preferred over logistic regression when dealing with complex data dependencies and temporal dynamics. They excel in handling sequential data, where relationships between inputs and outputs evolve over time, as they can model hidden state evolution and transitions. And, in contrast to SVR, HMMs exhibit greater flexibility in modeling complex temporal dynamics, especially when data includes multiple intertwined sources of variability that defy traditional regression models. Nonetheless, HMMs can be computationally demanding and necessitate ample data for effective training.

Before delving into deep machine learning methods, it is important to mention rule-based methods, used for occupancy prediction in regression analysis. Rule-based methods define a set of rules based on prior knowledge or data exploration for occupancy prediction. These rules typically take the form of 'if-then' statements, specifying conditions (if) and actions (then). For example, a rule-based method can be created by specifying conditions based on the time of day, temperature, and historical occupancy patterns, using these rules to predict future occupancy. Rule-based methods offer interpretable outcomes

comprehensible to non-experts. However, they often require significant manual effort for rule development and fine-tuning and may have limited performance with complex or noisy data. Consequently, they are sometimes combined with deep learning methods. Dorokhova et al. (2020) proposed a rule-based scheduling approach for commercial building air conditioning systems using occupancy forecasting. This approach merges machine learning-based occupancy prediction models with a rule-based scheduling system to optimize air conditioning system operation, underscoring the potential of uniting machine learning with rule-based systems for energy-efficient building automation.

1.3.3.2 Advanced regression methods based on networks

Traditional regression approaches, including time-series analysis, logistic regression, linear regression, SVR, HMM, and rule-based approaches, while effective for occupancy prediction, have inherent limitations. They rely on predefined assumptions, linear relationships, or manual rule crafting, making it challenging to capture intricate and nonlinear patterns in occupancy data, ultimately leading to suboptimal performance. To overcome these limitations and enhance occupancy prediction, researchers have turned to deep learning-based regression methods.

In fact, a large majority of the algorithms mentioned in the section on classification task (section 1.3.2) can be used for regression task. Their relative efficiencies vary according to the type of objective (classification or regression). While the literature on deep learning for occupancy and activity prediction is extensive, traditional ANNs still find utility. For instance, Chen et al. (2021) combined ANN with fuzzy logic to predict office building electricity demand across different occupancy rates, achieving promising predictive capabilities. Manno et al. (2022) employed a Shallow NN (SNN) approach to successfully forecast hourly electricity and heating demands in the short term, emphasizing computational efficiency and interpretability advantages compared to ARIMA, SVR and LSTM networks.

As already mentioned above, a summary table, outlining key occupancy prediction methods based on their features, applications, data types, complexity, strengths, and weaknesses, is provided in Appendix A (Table A.7). In this thesis, we explored a diverse range of methods listed in this Table A.7, but not all were thoroughly examined. Some were excluded due to specific data requirements, such as the CNN method for image

processing (even though sequential encoded data could be used with this method), which relied on camera data not available in our study. We aimed for a balance between methodological diversity and practicality, selecting methods aligned with our needs and suited to the nature and dimensionality of the data. For both supervised and unsupervised classification and / or regression techniques, we deliberately incorporated a range of method complexities in our analysis. For classification tasks, we compared unsupervised methods like PCA and DBSCAN (considered as basic methods) with the MGD and UMAP methods (respectively, an intermediate approach capable of handling correlated variables and an approach capable of capturing non-linear relationships and preserving complex structures within the data). For multiple activity detection, we combined ANN, PCA, Monte Carlo Tree Search (MCTS), and BN methods, exploring different complexity levels and employing graph-based and Deep Learning techniques. For regression in supervised settings, we explored both basic logistic regression methods and advanced ANN models. Detailed justifications for these method selections will be provided in their respective sections.

In addition, it should be noted that measured data from sensors are mostly used to run classification or regression algorithm. Although less frequently observed in the literature, it is also possible to train the algorithms on simulated data.

1.3.4 Some problems in using machine learning for activities and events detection, estimation/prediction

Occupancy estimation/prediction methods, particularly those using machine learning techniques, have limitations. In Table A.8 (Appendix A), we aim to summarize these limitations concisely. The first column of the table presents identified limitations, followed by explanations and their impact in the second column. The last two columns provide examples of methods that are most and least sensitive to these limitations. We have organized these limitations hierarchically based on their perceived impact, with the first two limitations considered to have a very important impact, the next three with a significant impact, and the last four with a moderate but still significant impact. The severity may vary depending on the specific context and data characteristics.

In this thesis, focused on accurate activities / events detection, we face central challenges stemming from data nature, often unlabeled and high-dimensional, and various algorithmic aspects, such as tuning parameters and overfitting. This section outlines key

challenges and implemented solutions.

Among the nine major problems presented in Table A.8 (Appendix A), our primary focus lies in tackling the issues of data scarcity and the need for labeled data. We aim to process data without pre-assigned labels, addressing real-world scenarios where manual data labeling is time-consuming and labor-intensive, potentially raising privacy concerns. To achieve this, we employ unsupervised and semi-supervised learning methods, as shown in Chapter 4 on Single activity detection, featuring as DBSCAN and AE. These methods identify patterns or anomalies in data without relying on labels. Another key objective related to these problems and parameter sensitivity is establishing universal principles for selecting tuning parameters and thresholds. In the same Chapter 4, we explore the impact of different parameters and thresholds on algorithm performance and develop visualization methods for different algorithms to establish reliable foundations for future work in the field. Building upon the foundation in addressing the primary challenge, we extend our work to the more challenging task of detecting multiple activities and events using unsupervised and semi-supervised techniques, which will be discussed in Chapter 5, focusing on multi-activity detection.

Other challenges will be addressed in this thesis, including overfitting, data non-linearities, and imbalanced data. Overfitting, along with its counterpart, underfitting, is a common issue in machine learning, arising from the bias-variance trade-off. Bias refers to the error resulting from model assumptions, potentially causing the algorithm to overlook relevant feature-target relationships (underfitting). Variance, on the other hand, pertains to errors due to model sensitivity to training data fluctuations, leading to poor performance on unseen data (overfitting). Finding the right balance between overfitting and underfitting is crucial. In Chapter 4, we will address these challenges employing techniques such as polynomial feature generation to adapt model complexity and regularization to penalize and prevent overfitting.

Another challenge arises from processing data with strong non-linearities, which can be seen as a sub-category or inherent feature of previously discussed challenges like overfitting, non-stationarity, and parameter sensitivity. Real-world data frequently exhibits complex non-linear relationships between variables, adding complexity to modeling and prediction. For instance, overfitting can be exacerbated by non-linear data as complex models tend to overlearn unique non-linear patterns from the training data. The abundance of features and strong non-linearity can overwhelm some machine learning algorithms, making it challenging to discern data patterns.

In Chapter 4, we will address this challenge by utilizing UMAP to visualize data patterns in lower dimensions. Additionally, we will compare UMAP to PCA, a traditional dimensionality reduction technique that simplifies data structure while preserving most variation.

Lastly, the challenge related to imbalanced data in real-world scenarios is addressed, particularly the class-imbalance problem between normal and abnormal classes in supervised activity detection. In cases like fault detection, anomalies are scarce compared to normal samples, making it difficult to learn effective rules with limited negative samples (outlier). Usually, classification machine learning tasks expect the samples size of each category to be balanced. In our study, the classes are strongly (and deliberately) imbalanced. In our analysis, we will first present the results without handling the imbalanced data, and then we will adopt two preprocessing methods for class-imbalance (refer to figures 1.3 and 1.4). The first method is under-sampling (Liu et al., 2009), which involves selecting a small number of samples at random from the majority class (in our study, the closed window state class). These samples are then combined with the original minority class samples to create a new training dataset. The second method is over-sampling, which expands the minority class by duplicating observations from that class.

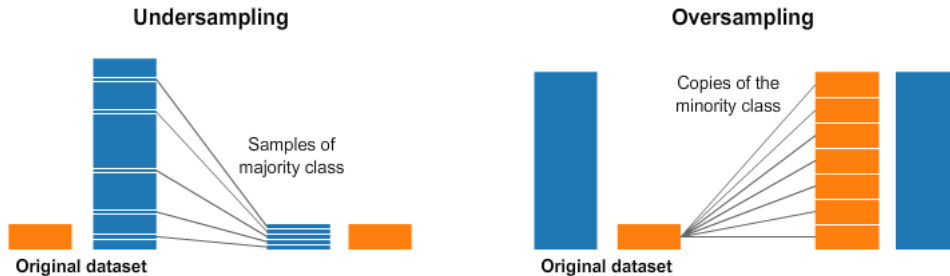


Figure 1.3 – Illustration of under and over -sampling techniques for imbalanced data.

While the random oversampling algorithm is the simplest method, it can lead to model overfitting as it makes the model too specific to the learned information. To address this, we will improve upon the method by using data synthesis to generate more data based on existing data. In Chapter 4, we will address this specific challenge by using logistic regression and ANN combined with under- sampling, over-sampling, and the Synthetic Minority Oversampling Technique - SMOTE - (Chawia et al. 2002) to balance the inputs. We also compare the effects of different imbalance handling techniques on various algorithms in Chapter 4, along with the analysis of the root causes of these effects.

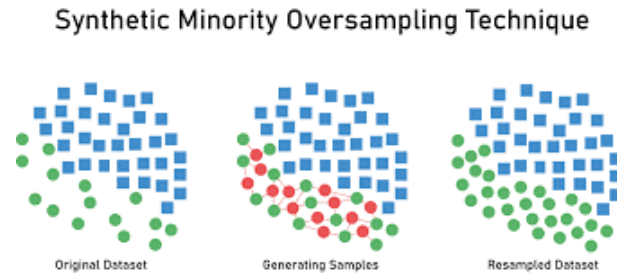


Figure 1.4 – Illustration of Synthetic Minority Over-sampling Technique – SMOTE - (Chawla et al., 2002)

1.4 Performance prediction and control of buildings

To achieve user-centric optimization in smart buildings, energy systems require effective control strategies. These strategies are important for optimizing building systems like, HVAC to improve both energy consumption and comfort. Traditionally, building thermal control methods rely on model-based approaches. However, model uncertainties and modeling errors always exist in the modeling process. Model effective control stands out as one of the most efficient model-based technique in building thermal control.

1.4.1 Control strategy in buildings

As already mentioned, buildings currently contribute to approximately 50% of global electricity consumption and around 25% of carbon dioxide (CO_2) emissions, with HVAC systems responsible for about half of this energy usage. To enhance energy efficiency and occupant thermal comfort in modern smart homes, advanced HVAC controls have been extensively explored, showing potential energy savings ranging from 13% to 28% (Gyalistras et al.2010; del Mar et al.2014; Roth et al.2002). Optimally, the full utilization of this technology could lead to annual energy savings of 8 to 18 petawatt-hours (PWh) (Drgona et al.2020). Understanding this potential, the European Union has mandated the installation of building automation and control systems in large buildings by 2025 (Europeancommission,2019), highlighting the significance of advanced control theory in future building energy management.

However, despite the substantial energy-saving potential, the majority of buildings still rely on basic rule-based control (RBC) methods, such as "if- then" rules, due to their simplicity (Kawakami et al.2014 ; Shakeri et al.2017). These approaches lack the predictive

capabilities of more sophisticated control strategies and may limit energy savings (Aghemo et al.2013 ; Mechri et al.2010). Other traditional control systems use "bang-bang" controllers or Proportional-Integral-Derivative (PID) controllers, which are straightforward but ill-equipped to optimize complex objective functions or plan ahead, leading to excessive energy consumption.

Further complications emerge due to uncertain and fluctuating disturbances. For example, if the system initiates heating in the morning on a sunny day, there is a high likelihood that - given the slow thermal dynamics and heat gains through windows - the temperature could surpass comfort limits if the controller does not halt heating in a timely manner.

A more advanced approach is Model Predictive Control (MPC), which uses a model of the system to anticipate future results and adjust control inputs based on forecasted states, rather than solely relying on present or past states, a significant improvement over the traditional "bang-bang" and Proportional-Integral-Derivative (PID) controllers. MPC also takes into consideration weather and occupancy forecasts. The OptiControl research project (Gyalistras D et al.2009) assessed MPC's potential for energy savings in building climate control, revealing its effectiveness and superior energy-saving potential compared to other control theories (Oldewurtel et al.2010 ; Oldewurtel et al.2012).

Importantly, MPC consistently integrates occupancy information into the control process, simplifying the comparison of various occupancy data types for ideal building control based on occupancy forecasts. This approach works independently of parameters or threshold adjustments required in other control theories.

1.4.2 Importance of occupancy centric control

As previously mentioned, the MPC framework consistently incorporates occupancy information into control, and it is crucial to understand why this inclusion is so important. Klepeis et al. (2001) found that people spend over 87% of their lives indoors. Building energy consumption is influenced by various factors, including engineering innovations, cultural norms, occupancy behavior, and social justice concerns. Occupancy behavior encompasses the number of people in a space, their interactions like adjusting windows, blinds, and lights, and their preferences for temperature and illumination. Occupants also affect the indoor climate by generating heat, CO₂, and using energy-intensive devices like laptops, which contribute to internal heat gains and power consumption. These dynamics significantly impact a building's thermal behavior, making occupancy data essential for

accurately simulating building dynamics and ensuring comfort levels throughout the day.

While much research focuses on specific aspects of building management, such as demand response, energy cost reduction, and thermal comfort, there is a notable lack of studies in occupant-centric control. Many prioritize energy savings over occupant comfort, even though earlier research has explored HVAC systems' potential for demand response and the impact of occupant presence on energy use. Understanding individual occupants' preferences is important for their well-being and satisfaction, reducing conflicts between residents and building energy systems, ultimately improving both energy efficiency and occupant comfort. The integration of occupant behavior into building control can significantly enhance both aspects. To address occupant-oriented demand response, various techniques, including occupancy schedules, building models, and data sources, have been identified. Some studies, like those by Jin et al.(2017) and Biyik et al.(2019) employed Resistor-Capacitor (RC) models with thermal parameters from actual building data, ensuring reliability and applicability for Model Predictive Control (MPC). However, these studies did not incorporate an occupancy schedule into their MPC algorithms. In contrast, studies like Hu et al.(2019) used attendance schedules but relied on artificial data from TRNSYS simulations.

An occupant-centric MPC strategy can be deployed to reduce energy expenses while increasing occupant comfort. Upon detecting occupants, one approach is to employ a feedback controller for adjusting lighting and ventilation promptly. However, thermal dynamics can be slow, and heating or cooling systems may require several hours to return to a comfortable temperature after an adjustment.

Hence, forecasting future occupancy is critical. Another option is a predictive controller with a flexible occupancy schedule to maintain temperature limits over time, ensuring comfort for early and late occupants. However, it may become less effective if tenant behavior or room assignments change. Studies have shown substantial energy savings potential with occupancy prediction-based cooling control, ranging from 7% to 52% compared to conventional systems (Peng et al.2018). Incorporating occupancy data into MPC represents a paradigm shift in building energy management. While traditional methods have merits, MPC with predictive capabilities effectively optimizes energy use and occupant comfort. Success depends on accurate occupancy forecasting and the building's thermal model complexity. Future research could explore machine learning algorithms to enhance occupancy predictions, improving MPC's effectiveness.

1.5 Conclusion of this chapter

This chapter extensively reviews smart building energy management literature, covering topics that include performance metrics, demand-side regulation, sensor placement, communication technologies, occupants activities, occupancy monitoring, detection methods and building control algorithms. It begins by highlighting the critical role of smart houses in the global energy optimization, addressing current challenges, and emphasizing user-centric control's significance. Key issues, including sensor placement and non-intrusive multi-occupant activity detection, are discussed. The chapter compares various communication systems, sensors, optimal sensor location algorithms, and user behavior categories. It also evaluates methods for detection, identification, and estimation. Identifying these challenges and research gaps not only underscores the importance of this study but also serves as a guiding framework for methodological choices in this and future study. In subsequent chapters, the chapter 3 will address the issue the placement of sensors using the information-based and statistical method whom choice is motivated by the robustness and information independence characteristic of the method. In chapter 4, we will address the issue of occupancy activity detection with considering the occupancy privacy by using ML algorithm whom choice is motivated by the model performance, robustness and the degree of label data demand. In chapter 5, we will address the issue of multi-activities detection under the premise of respect the occupancy privacy using two hybrid method.

PRESENTATION OF THE CASE STUDIES

The developments of this thesis will be applied to two case studies in the following three chapters. The purpose of this chapter is to describe these two case studies. The first is a simulated case study. Dynamic building energy simulations (DBES) are performed on a virtual building consisting of one thermal zone. CFD simulations are then performed to obtain information about the spatial distribution of temperature fields in the room. The second case study concerns a real educational building, in which two classrooms were instrumented during the thesis to collect data on occupancy and indoor comfort.

2.1 Introduction of the chapter

The aim of the thesis is to improve the energy management of buildings by better taking into account the occupant's actions and preferences. To this end, several methods will be investigated in the following three chapters to optimise the placement of multiphysical sensors to detect one or more occupant's actions that will be used to develop optimal user-centric management strategies. The consistency of the proposed general framework will be demonstrated by applying the methods to the case studies presented hereafter in this chapter. Data on buildings and occupancies are required to train and test the algorithms included in the thesis framework. The originality of the project is to use both virtual and real data. Therefore, two cases have been studied: a simulated case study and a real one. Firstly, a fictitious case study has been developed using simulation softwares. The building modelling allows to perform both dynamic building energy simulations (DBES) and CFD simulations. It can therefore be used to estimate indoor temperatures and energy loads for different predefined occupancy scenarios. The building, the software, the modelling assumptions and the first results are presented in the second section of this chapter (§ 2.2). Secondly, as the methods were applied to the first case study, an instrumentation has been designed and deployed in a real building as part of the thesis project¹. The smart

1. Project RFI-Wise Building Internet of Things (BIoT)

multiphysical sensors installed allow the collection of data on the indoor environment, as well as on occupancy. In the third section of this chapter (§ 2.3), the instrumented building is presented and the solution designed to collect the data is explained.

2.2 Simulation-based case study

2.2.1 Aim of the case study

The methods in the thesis framework are first applied to virtual building data obtained using DBES and CFD simulations. As pointed out in the previous chapter, machine learning and statistical methods have been less frequently used to handle simulated building data. However, it has several advantages. Firstly, occupancy data can be collected easily, cheaply and non-intrusively, as there is no need to install sensors. Secondly, several occupancy and climate scenarios can be studied on a building during the same period of time. The ability of machine learning methods to detect the actions of occupants in different circumstances can then be tested. Therefore, the use of virtual building data is an interesting option prior to the deployment of sensors in a new or existing building. Furthermore, it can be useful for finding the best locations for sensors installation, as well as for evaluating the effectiveness of energy management strategies. Finally, dealing with virtual data is an appropriate way to obtain well-calibrated models to detect faults in real systems, as shown by the concept of *in situ* virtual calibration (Yoon et Yu 2017; Yoon 2020), which involves training a model using virtual data first, and then improving it based on the data collected by the sensors. However, using virtual data can be challenging. The building being assessed must be accurately modelled, while the physical properties of existing buildings are usually unknown. Realistic occupancy scenarios must be defined in the simulations. Additionally, multiphysical simulations have to be performed to get an overview of the indoor ambient conditions.

In this work, the simulation-based case study is considered as a preliminary step to the real-data-based case study. While the statistical methods were studied and deployed on the virtual case study, the monitoring of the real building described in the second case study was prepared. In this preliminary step, a fictitious single-zone building, consisting of one room, is considered. In addition, the indoor temperature is the only physical quantity assessed.

2.2.2 Modelling tools

The DesignBuilder² software was chosen as the modelling tool to simulate the dynamics of heat transfer in the building. It has the advantage of integrating both DBES and CFD simulations in one software. DBES are necessary in this work to obtain the evolution of the air temperature in each thermal zone. The changes in the temperature profile are then analysed to identify the user's actions in Chapter 3 and Chapter 4. CFD simulations provide additional information: temperatures can be computed at different positions in the same thermal zone. Air temperatures at several points near the wall and at specific locations (where occupants are more likely to be) are used to select the optimal set of sensors to identify user's actions and comfort in Chapter 3.

The DBES engine of DesignBuilder is the well-recognised open-source program EnergyPlus³, developed by the National Renewable Energy Laboratory (NREL, US), as well as other research laboratories and private companies. In EnergyPlus, heat and mass balances are performed at user-definable time steps on all thermal zones of the modelled building. The temperature in each thermal zone is then obtained and the heating or cooling loads can be computed. In addition, HVAC systems can be described to calculate energy consumptions. The reliability of the results given by EnergyPlus has been demonstrated in the past by experimental comparison (Spitz et al. 2013) and model inter-comparison (Judkoff et Neymark 1995; 2013; Brun et al. 2013).

The CFD engine developed by DesignBuilder is used to perform CFD simulations. The boundary conditions (surface temperature, heat sources, and airflow rate) are entered into the CFD engine based on the EnergyPlus results. The spatial distribution of the temperature fields in a thermal zone of a building is obtained, taking into account the effect of HVAC systems. The indoor thermal comfort can thus be studied. The validity of the engine has been tested in the past by Northumbria University⁴ against the CFD software Phoenics⁵, which in turn has been validated by comparison with experimental results and theoretical analyses.

2. <https://designbuilder.co.uk/>

3. <https://energyplus.net/>

4. <https://designbuilder.co.uk/cfd>

5. <https://www.cham.co.uk/phoenics.php>

2.2.3 Building description

A fictitious building is modelled in DesignBuilder. It consists of one room, which is considered to be an office building, with a floor area of 120 m^2 . The room has a rectangular shape of $12\text{ m} \times 10\text{ m}$, with a 3.5 m height. The main façades face north and south respectively. Three large windows open on the north façade. Two smaller windows and a door open to the south. An image of the building modelling is given in Figure 2.1.

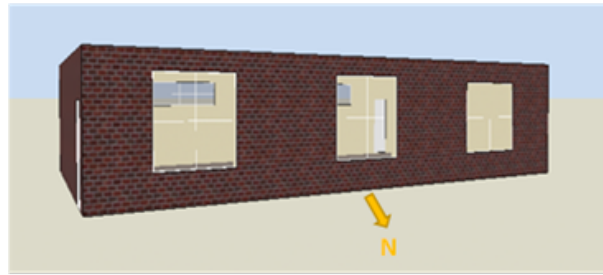


Figure 2.1 – Fictitious building modelling in DesignBuilder.

The building is a partially refurbished building. An external insulation is considered. Windows U-value is $2\text{ W/m}^2/\text{K}$ and a solar factor of 0.62 . The external walls and the roof are in contact with the exterior and the rooms' floor is in contact with the ground. The wall compositions are given in the following three tables. It is assumed that the building is heated by a gas boiler and has natural ventilation.

Table 2.1 – External walls composition

	Thickness (cm)	λ $\frac{\text{W}}{\text{m}\cdot\text{K}}$	ρg $\frac{\text{kg}}{\text{m}^3}$	CS $\frac{\text{J}}{\text{kg}\cdot\text{K}}$	R $\frac{\text{m}^2\cdot\text{K}}{\text{W}}$
Cement mortar plaster	1	0.72	1,760	840	0.01
XPS extruded polystyrene-CO ₂ blowing	20	0.034	35	1,400	5.88
Concrete, cast-heavy weight, moist	16	1.7	2,000	840	0.09
Gypsum plastering	1.3	0.4	1,000	1,000	0.03
				Wall	6.02

Table 2.2 – Roof composition

	Thickness (cm)	λ $\frac{W}{m \cdot K}$	ρ $\frac{kg}{m^3}$	CS $\frac{J}{kg \cdot K}$	R $\frac{m^2 \cdot K}{W}$
Sealing layer	1	0.7	2,100	1,000	0.01
XPS extruded polystyrene-CO ₂ blowing	25	0.034	35	1,400	7.35
Concrete, cast-heavy weight, moist	15	1.7	2,000	840	0.09
Air gap	2	-	-	-	0.18
Plasterboard	1.3	0.25	2,800	896	0.05
				Wall	7.69

Table 2.3 – Floor composition

	Thickness (cm)	λ $\frac{W}{m \cdot K}$	ρg $\frac{kg}{m^3}$	CS $\frac{J}{kg \cdot K}$	R $\frac{m^2 \cdot K}{W}$
Sealing layer	1	0.7	2,100	1,000	0.01
XPS extruded polystyrene-CO ₂ blowing	20	0.034	35	1,400	5.88
Concrete, cast-heavy weight, moist	20	1.7	2,000	840	0.12
Timber flooring	3	0.14	750	1,200	0.21
				Wall	6.23

2.2.4 Building energy modelling

2.2.4.1 Assumptions

The modelled classroom is built as a monozone building: only one air temperature will be computed for this room. The simulation period is one year from first of January with a 15 minutes time step. The weather file for Nantes, available in DesignBuilder, has been linked to the simulation. The degree-hours⁶ reach 56,400 for this location. The building airtightness was set to a constant rate of 0.15 air change per hour.

The following use scenarios were assumed. The heating setpoint is 22 °C, the heat set back point during nights is 18 °C. Occupants are assumed to be in the building on working days during office hours, i.e. between 9 a.m. to 7 p.m. on weekdays. There is no occupancy

6. The degree-hours are a metric to quantify the energy loads. For each hour of the year, the temperature differences between the outdoor temperature and a threshold are summed. For instance, on January 1st between midnight and 1 a.m., if the outdoor temperature is 3 °C and the heating threshold temperature is 18 °C, then $18 - 3 = 15$ °C are registered; the same thing is done for each hour of the heating season for which the outdoor temperature remains below the threshold. A high value for the heating degree-hours indicates a cold climate.

during weekends, from 15th December to 15th January, and in July and August. The occupied duration is randomly chosen between 1.5h to 7h. Up to ten people can be in the building. In order to consider a diversity of building use, an occupants' schedule has been randomly generated. Each day, a different occupancy is considered, as it is assumed that the occupancy may vary considerably in this building. In the simulation, occupancy is modelled using an internal gain scenario describing metabolic rates. It is assumed that one occupant generates 123 W^7 . The light and the working equipments are assumed to be on when occupants are in the building. They provide additional internal gains. An example of a weekly occupancy schedule is shown in Figure 2.2

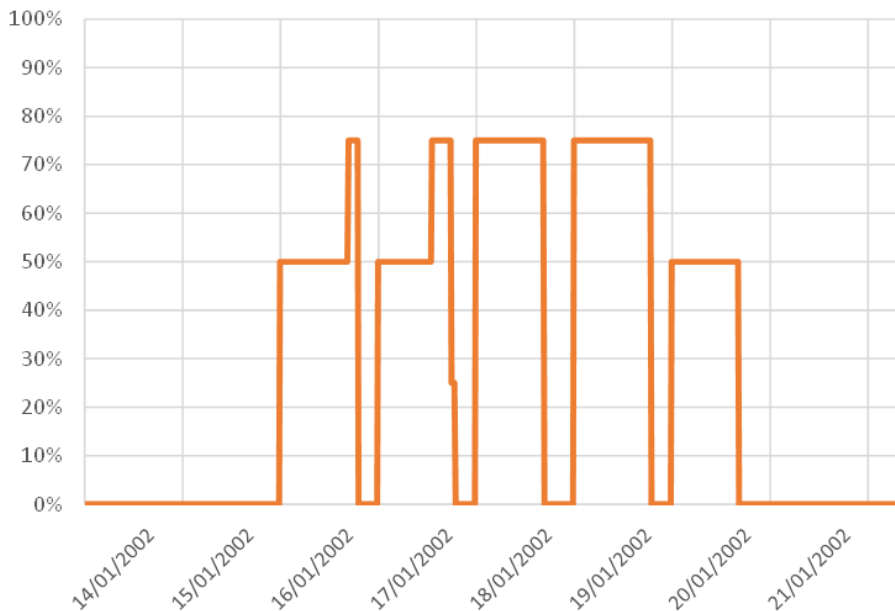


Figure 2.2 – A week occupancy schedule from the generated scenario.

One occupant action has also been modelled: actions on windows. Two window opening schedules were randomly generated considering some constraints: window opening can only occur between 8 a.m. to 7 p.m. on weekdays, window status can change every 15 minutes, and windows remain opened for up to 30 minutes. In the next chapters on occupants' action detection, window opening will be recorded as 1 and the closing as 0. In order to study the ability of the machine learning algorithm to detect rare events, highly imbalance classes were generated. In the first randomly generated schedule, only 147 cases of window opening were assumed among the 35,040 time steps simulated. The window

7. Metabolic rate per person for light office work

opening rate in the whole datasheet is thus about 0.0042. A second more realistic window opening schedule has been randomly generated with 993 window openings over the year. The window opening rate is then 0.028.

2.2.4.2 Example of results

The DesignBuilder simulation allows to obtain the evolution of the temperature in each thermal zone throughout the simulated year at an interval corresponding to the fixed time step. This gives an idea of the thermal comfort within each zone. Based on this result and on the defined setpoints, the heating or cooling loads are obtained, indicating the building energy performance. For this case study, each simulation required 3 min of calculation in the software. Considering the baseline case study, the yearly heating loads reached 26.8 kWh/m^2 , which correspond to a good energy performance⁸. The evolution of the temperature in the room is given in Figure 2.3, with the corresponding outdoor temperature. Windows opening from the first scenario are marked with a green cross on the indoor temperature points.

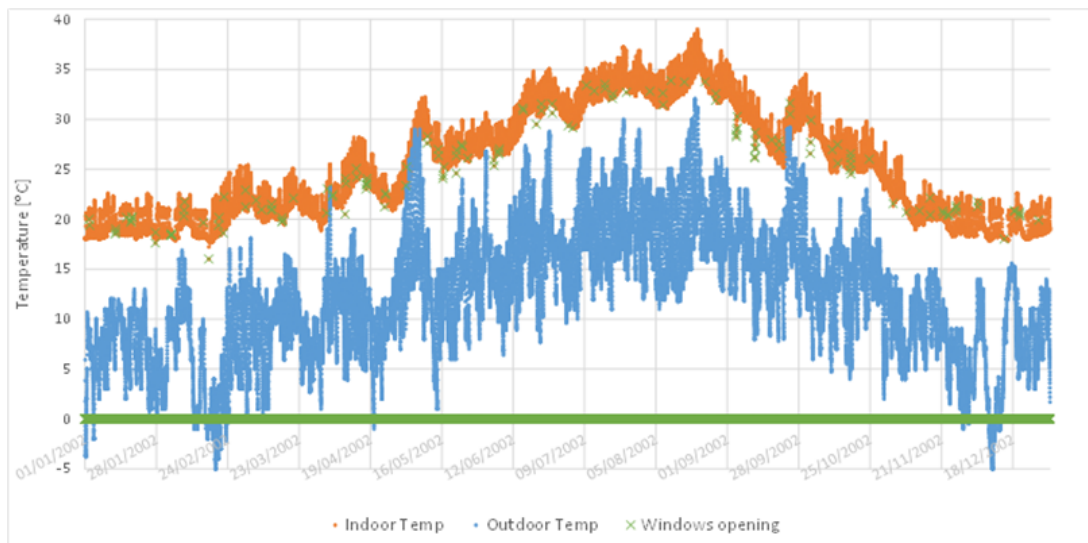


Figure 2.3 – Evolutions of the indoor and outdoor temperatures during the simulated year.

In winter, the temperature oscillated between the heating setpoint and setback point. As no summer comfort strategies have been modelled (e.g. night free cooling, use of

⁸. The heating load is quite close to that required for the very energy-efficient Passivhouse standard (15 kWh/m^2), whereas the average heating loads for French buildings is about 200 kWh/m^2 .

blinds), the temperature increases a lot in summer. Openings often correspond to an indoor temperature drop as the outdoor temperature is generally lower than the indoor one.

2.2.5 Building CFD modelling

2.2.5.1 Assumptions

For a few days distributed over the year, CFD simulations are performed. On these days, one simulation is done every hour. The information from the energy simulation of the given hour is imported and used as boundary conditions for the CFD simulation. Among this information, the air inlet temperature is taken from the weather file and wall temperature are calculated in the energy simulation. Then the building is meshed into a series of interconnected cells. Each cell is assigned a set of properties, such as temperature, pressure, and airflow rate, which are solved using a set of partial differential equations.

2.2.5.1.1 Grid resolution In the CFD, higher grid resolution means that the geometric domain is divided into smaller cells, which allows for more accurate representation of the fluid flow properties within each cell. It can better capture flow phenomena, such as boundary layers, turbulence, and complex geometries. However, a larger grid with more cells requires more memory and processing power, which can significantly increase the simulation time. So adjusting the grid size and resolution is a trade-off between accuracy and simulation speed. Designbuilder has the adaptive grid refinement techniques which can automatically refine the grid in regions where higher resolution is needed, such as near walls or areas with complex flow phenomena. This allows for more accurate simulations without significantly increasing the overall grid size and computational cost. In this study, the default adaptive grid resolution was used, resulting in the grid shown in Figure 2.4.

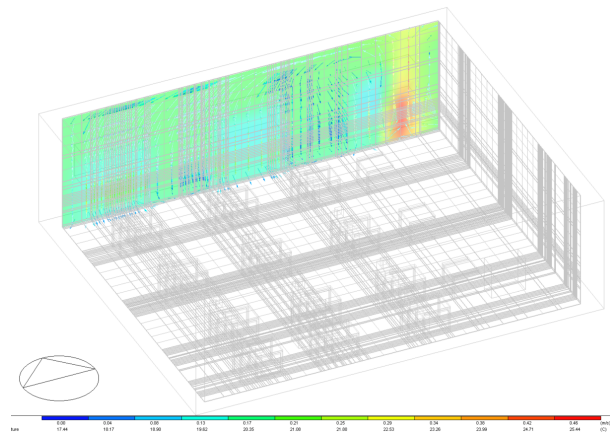


Figure 2.4 – Grid resolution.

2.2.5.1.2 Solver settings DesignBuilder uses a finite volume method that involves subdividing the calculation domain into non-overlapping adjoining rectilinear volumes or cells. The differential equations are then converted into a set of linear algebraic equations for each cell. The equations are solved using an iterative scheme, accounting for the nonlinearity of the equations through nested iterations. The outer loop iterates until the solution converges. Convergence in this context means that the values of the dependent variables satisfy the finite difference equations for all the cells. However, because the coefficients of the equations contain variables dependant to each other, convergence isn't always guaranteed. To help achieve convergence, DesignBuilder uses false time steps, which are part of a pseudo-transient formulation. This formulation effectively slows the change in dependent variables, leading to a more stable solution. The following convergence criteria are considered in DesignBuilder:

- Termination Residual: sets the maximum acceptable change in the dependent variables between consecutive iterations. The simulation will be considered converged when the changes in all dependent variables are less than the specified tolerance.
- Maximum iteration: sets the maximum number of iterations that the solver will perform before stopping. If the simulation does not reach the desired tolerance level within the specified maximum iterations, the solver will stop and report a non-converged solution.
- Under-relaxation factors: help control the rate of change of the dependent variables during the iterative process. By adjusting these factors, it is possible to influence the stability and speed of the convergence process. Lower values can lead to slower but more stable convergence, while higher values can speed up the process at the

risk of instability.

In the CFD computations performed for the case study, the termination residual is set as 0.00001, the maximum iterations is 5000, the under relaxation factors is 1. Setting the Termination Residual to 0.00001 indicates that a high degree of accuracy is desired for the solution. The lower the termination residual, the smaller the allowed discrepancy between the current and desired solutions. This means that the iterative process will continue until the residuals of the dependent variables for all cells in the domain are less than 0.00001. In other words, the solution must be very close to the true result before the process is considered to have converged. This may lead to longer computation times but higher accuracy. Setting the Relaxation Factor to 1 implies that there is no under-relaxation, and the full calculated value of the current iteration dependent variable is assigned to the variable. This means that the iterative process will progress at its full speed without damping the updates in the dependent variables. In some cases, this can result in faster convergence; however, it might also lead to instability or divergence in certain situations. The choice to set the Termination Residual to 0.00001, the Relaxation Factor to 1 are desire for a high degree of accuracy, a relatively fast convergence, and the maximum iteration 5000 is for time control, the computation will stop at 5000 iterations, even if the solution has not yet reached the specified Termination Residual of 0.00001.

2.2.5.2 Example of results

Once the input data for the CFD simulation are defined, the simulations are launched. Results can be extracted in form of air velocity and temperature for different slices of the building, allowing a better understanding of convection phenomena at specific locations in the room. Each CFD simulation required 15 min of computation. An example of results is given in Figure 2.5 for January 1st at 12 am. The air velocity and temperature are given for two slices of the building.

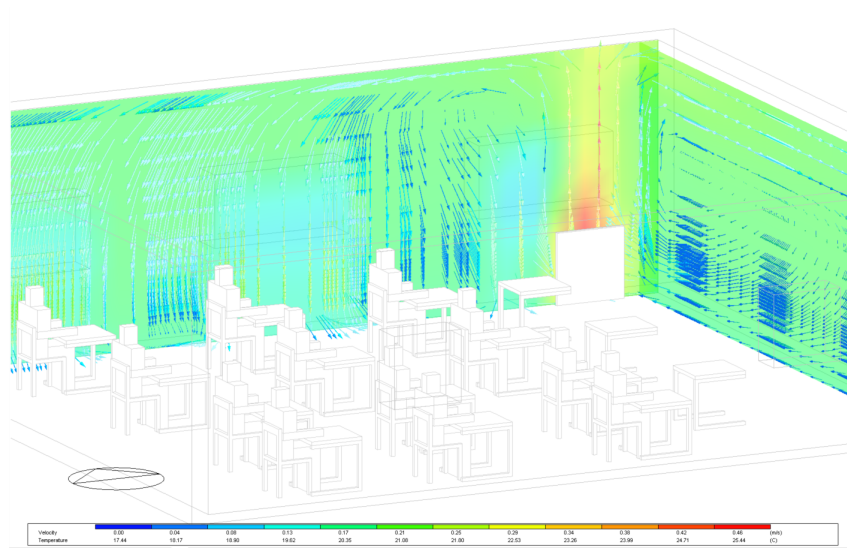


Figure 2.5 – CFD simulation results for January 1st at 12 am.

For every selected slice, the air velocity and the air temperature can be extracted at a spatial interval of 30 cm. For the chapter 3 on optimal sensor location, temperatures are extracted from slices as close to the four external walls as possible (at a distance of 15 cm from the walls, because the sensors are assumed to be placed on the walls. Each extracted temperature is assumed to correspond to one possible sensor location. An additional slice is extracted at 1 m height in order to get the temperature at possible locations in the room, at some places where sitting occupants are assumed to be. The idea is to assess the comfort at these places.

2.3 Real data based case study

2.3.1 Aim of the case study

In contrast to the above-mentioned case study, the second one does not rely on simulated data but on real data collected in a building. Thus, the suitability of the methodology is investigated for a real application where data interpretation is more challenging because i) data may be missing due to various recording problems; and ii) occupancy information may be unknown (e.g. unrecorded presence) or inaccurate (e.g. difference between the actual and the reported number of occupants). However, processing real data offers new possibilities: instead of only processing temperature and heating loads, other physical values can be recorded, such as humidity, CO₂ concentration, or noise. This will potentially

facilitate the detection of occupant activities and the study of OSP.

2.3.2 Presentation of the monitored building

Data are collected in two classrooms of Polytech Angers Engineering School (which hosts the LARIS laboratory) of the University of Angers. A specific instrumentation has been designed and deployed in this building as part of the RFI Wise BIoT⁹ project, granted by the French Region of Pays de la Loire, in which the PhD thesis was conducted. The university building is located in Angers, 62 avenue de Notre Dame du Lac. It was built in the 1950s as a residential building, before being completely refurbished in the early 1990s for conversion into an educational building. Two extensions were added, one in the early 2000s and the other in 2014. The total surface of the school is 7,383 m² and is spread over five floors. The general shape of the building is rectangular with the main façades facing the north-east and the south-west, as shown in the site plan in Figure 2.6.

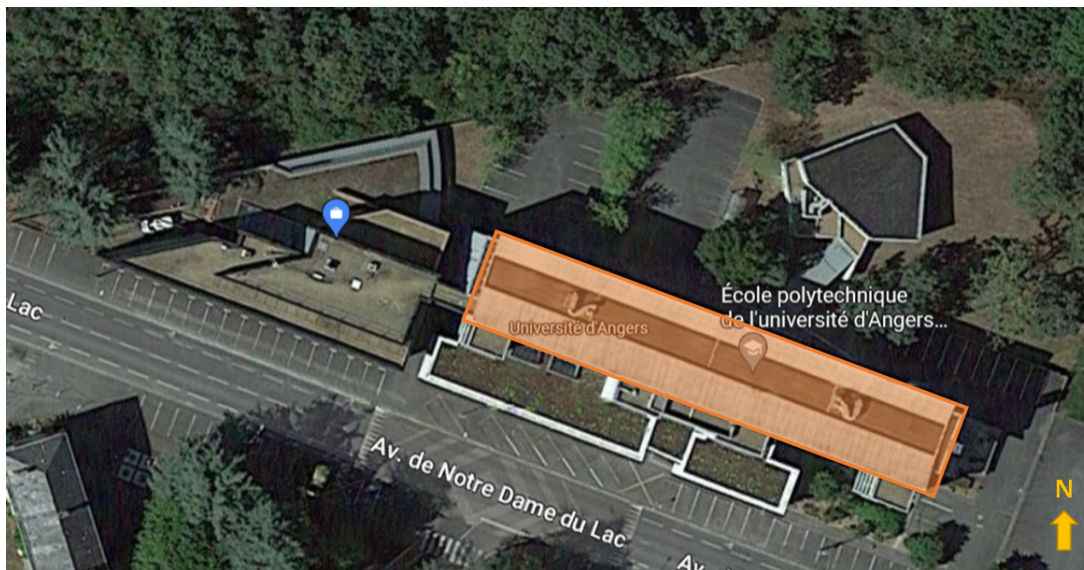


Figure 2.6 – Site plan of Polytech Angers. Adapted from Google Maps.

Two rooms have been selected to be monitored. They were chosen because they were among the most used classrooms during the academic year 2020-2021 according to the room booking tool of Polytech Angers. In addition, they both have north-east facing windows. In order to be representative of different teaching mode, the following rooms were selected:

9. <https://laris.univ-angers.fr/fr/projets/projets-anterieurs/biot.html> and <https://biot.u-angers.fr/>

- Room 114, located on the 1st floor, is a conventional lecture room of about 51 m². The only wall to the outside contains three windows. On the opposite wall, a window and a door open onto a corridor. The other two doors open to other classrooms. A picture and a sketch of room 114 are shown in Figure 2.7.
- Room 219, located on the 2nd floor, is a computer room of about 52 m² used for exercises and projects. As in room 114, the four windows are north-east facing. Two doors and a window open onto the corridor. The two remaining walls are in contact with other heated rooms. A picture and a sketch of room 219 are shown in Figure 2.8.

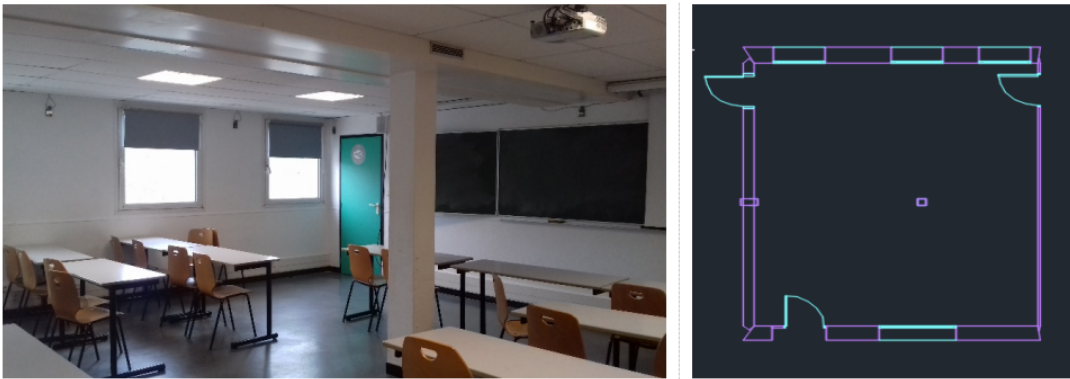


Figure 2.7 – Picture and sketch of room 114.

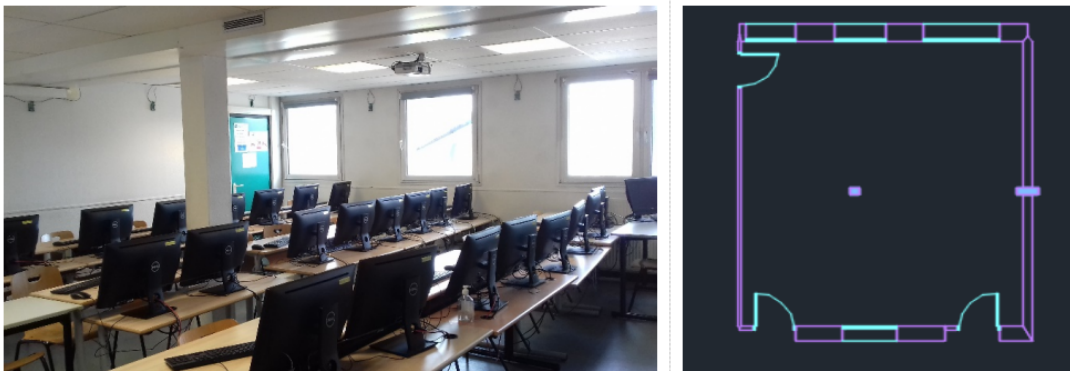


Figure 2.8 – Picture and sketch of room 219.

The two monitored rooms are located in the historical part of the building, highlighted in orange in Figure 2.6. For this oldest part of the building, the composition of the walls is unknown, hindering the possibility of modelling the same rooms in a simulation software in a simple and accurate way. Methods could have been used in order to assess the heat

transfer coefficient (Juricic et al. 2022), or to calibrate the model from measurements (Robillart 2015), but this is beyond the initial scope of this thesis.

2.3.3 Description of the monitoring

In this work, one of the objectives will be to optimize the placement of multi-physical sensors to detect one or more occupant's actions. It is intended to detect user's actions using a set of low-cost and non-intrusive sensors. The use of cameras and microphones has thus been discarded. A solution based on ambient sensors was preferred. Therefore, in the frame of the BIoT project, a specific instrumentation was developed; it is presented in § 2.3.3.1. In order to investigate the increase in the prediction quality when complementary measured data are used, a meteorological station was installed on the building (§ 2.3.3.2). Finally, labelled data on occupant's actions is required to train the algorithm. The set of real and soft sensors described in § 2.3.3.3 is used for this purpose.

2.3.3.1 Indoor ambiance measurement

In order to detect user's actions (see chapters 4 and 5) with low-cost and non-intrusive sensors, the instrumentation described in the following paragraphs was developed. It relies on the use of sensors measuring information about the indoor ambiance, such as temperature, humidity and CO₂. As the optimal placement of sensors will be investigated in chapter 3, a first typology of multiphysical ambient sensors was placed on the walls all around the two instrumented rooms. Then, a second typology of multiphysical sensors has been installed at some student's location. The data provided by this second typology will be used in chapter 3 to describe the comfort at the user's location, based on the optimal set of sensors on the wall.

2.3.3.1.1 Multiphysical sensors on the walls The first typology of multiphysical ambient sensors consists of a printed circuit board on which various commercially available sensors are welded. The printed circuit board, the deployment in the rooms and the data collection strategy have been designed by Alain Godon, associate professor at Polytech Angers, involved in the BIoT project. All members of the BIoT project were involved in the production of multiphysical ambient sensors: welding of individual sensors onto the boards and installation them in the rooms. Pictures of the assembly workshop and of a finished board are available in Figure 2.9.

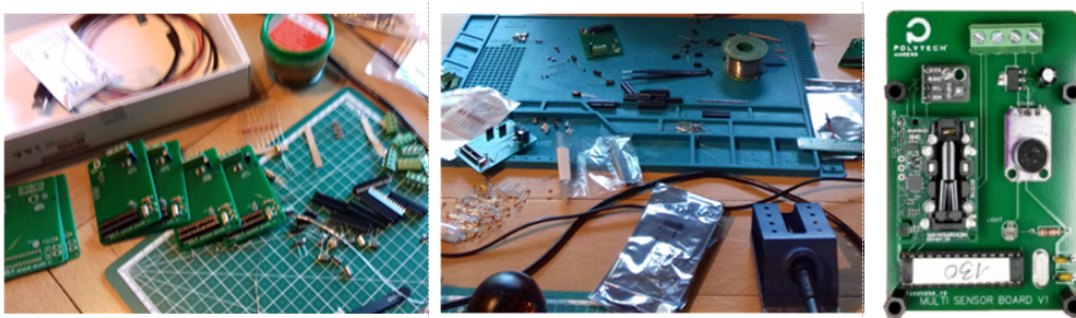
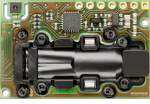
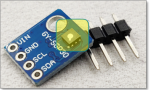

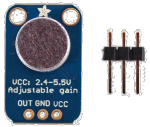


Figure 2.9 – Multiphysical sensor boards' assembly workshop and completed board.

Each board consists of four sensors able to measure the six values listed in Table 2.4. Temperature and humidity are natural quantities for assessing the occupants' comfort and their measurement can be performed with low-cost sensors. As mentioned in chapter 1, CO₂, noise and light have proven to be useful indicators for detecting the presence of occupants and eventually their actions. More expensive sensors are necessary for reliable measurement of CO₂. In addition, total volatile organic compounds (VOC) sensor were added to complement the measurement of CO₂ concentration, to provide information on the indoor air quality of the rooms. VOC sensor are often used as low-cost CO₂ sensors, but they do not directly measure CO₂ concentration. The data for all sensors is measured every 2 seconds and stored by an ATMEGA 328P microcontroller on the board. The microcontroller then averages the data over 60 seconds before transmitting it via a wire connection.

Table 2.4 – Measurement characteristics of each ambient sensor

Sensor	Measured value	Unit	Resolution	Range	Accuracy
Sensirion SCD30 	Temperature	°C	0.1	-40 to +70	± 0.4
	Humidity	%	1	0 to 95	± 3
	CO ₂	ppm	1	0 to 40,000	± 30
Sensirion SGP30 	Total VOC	ppm ²	0.2%	0 to 1,000	15% (ethanol)
Luna NSL-19M51 	Light	V	4.9mV	0 to 5 V	N/A
GY-MAX4466 	Noise	V	4.9mV	0 to 5 V	N/A

Some sensors were not calibrated at the factory. Calibration was therefore required. For the CO₂ concentration sensor, calibration was performed by placing the sensors for more than one hour in a highly ventilated room, so that the CO₂ level inside the room became homogeneous and reaches the outdoor CO₂ concentration. A calibrated CO₂ concentration sensor (Class'Air¹⁰) was placed in the same room. Its measurement was taken as a reference value and a correction was applied to the measurement of the other sensors to make them consistent with the reference value. The noise sensors were subjected to a calibrated sinusoidal signal and the gain of the GY-MAX4466 modules was adjusted to obtain the same output on all sensors. The important thing was not to obtain a calibrated value in a given unit such as decibels, but to obtain consistent values between the different sensors for differential analysis.

The multiphysical ambient sensors boards are placed approximately every two meters in the rooms. They are placed closed to the ceiling to avoid being accidentally damaged by students. 12 boards were installed in room 114 and 14 boards in room 219. Sensor placement is shown in Figure 2.10.

10. <https://pyres.com/solutions/classair/>



Figure 2.10 – Multiphysical sensor locations in room 114 (a) and 219 (b).

All boards are connected to each other, forming a sensor daisy chain. The four wires connecting all boards run through the false ceilings. In the chain, two wires are used for power supply and the other two for data transmission. For every three or four sensors in the chain, a microcontroller installed in the false ceiling collects the measured data by querying the boards the one after the other. Then, the microcontrollers send the data via Wi-Fi to the servers of the University of Angers. The measurements are accessible from the BIoT website¹¹. The recording started in February 2022 but the first two months of monitoring corresponds to a test and adjustment phase. Thus, reliable data were measured from May 2022.

2.3.3.1.2 Multiphysical sensors at students' locations The second typology of multiphysical sensors has been installed at some student's locations in the classroom. Each board measures temperature, humidity and CO₂ concentration, using the Sensirion SCD30 sensor, which is also installed in the first typology. Three boards have been installed under the student's desk in room 219 only and they are plugged on sockets for their power supply. These boards were not installed in room 114 as student's desk are movable in this room, and the boards were more likely to be disconnected. The data is collected every 10 minutes since the end of September 2022 and is sent via Wi-Fi to the university's servers.

¹¹. For instance, data collected in room 219 in October 2022 is accessible from this address: <https://biot.u-angers.fr/data/s219/2022/10>



Figure 2.11 – Multiphysical sensor board at student’s locations in the classroom 219.

2.3.3.2 Outdoor condition measurements

Besides monitoring indoor conditions, information on outdoor conditions are collected. This additional data can help to improve the quality of the user’s actions detection. For instance, a long period of absence outside the heating period could be identified if the indoor and outdoor temperatures follow each other for a long period. In addition, if windows are opened for a long period of time, the temperatures given by the sensors near the windows are more likely to be similar to the outdoor temperatures. The outdoor conditions are measured using the Vantage Pro2 weather station, from Davis Instruments, installed as part of the BIoT project on the roof of Polytech Angers (see Figure 2.12 a).

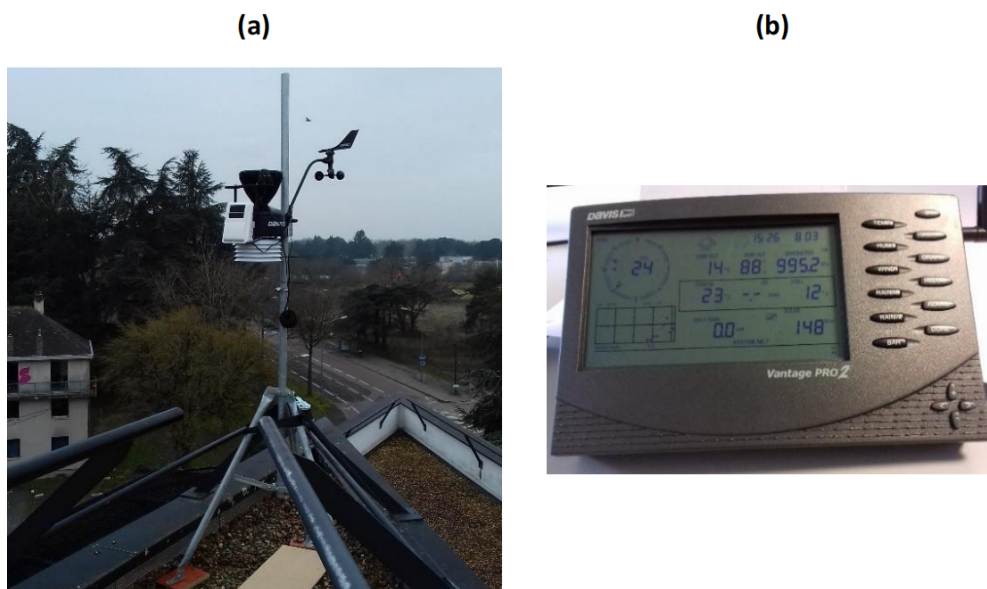


Figure 2.12 – Weather station installed on the roof (a) and data extraction module (b).

Various values are measured with this weather station: temperature, humidity, pressure, solar radiation, wind speed and direction. The accuracy of the measurements is shown in Table 2.5.

Table 2.5 – Measurement characteristics of the weather station, from the provided documentation

	Unit	Resolution	Range	Accuracy
Temperature	°C	0.1	-40 to +65	± 0.3
Humidity	%	1	1 to 100	± 2
Pressure	hPa	0.1	540 to 1,100	± 1.0
Solar radiation	W/m ²	1	0 to 1,800	5%
Wind speed	km/h	1	1 to 320	5%
Wind direction	°	1	0 to 360	3

The data is sent in real-time to a console (see Figure 2.12 b), which stores all measurements. The console must be regularly connected to a computer in order to empty the console’s internal storage card, access the recorded data and analyse it. Since February 2022, meteorological data are recorded every 5 minutes on the Polytech Angers building. As the weather station is provided by a company, no calibration of the sensors has been carried out.

2.3.3.3 Occupancy detection

Labelled occupancy data are required to train the algorithm to detect occupants’ actions. Therefore, specific sensors have been installed and some information is collected from additional sources (later called “soft sensors”). This allows for the identification of three kinds of user-related activities: actions on windows, use of electrical devices and presence.

2.3.3.3.1 Windows and doors openings detection Sensors have been installed on the windows and doors to report any change in state: opening or closing. The Shelly sensors have been selected for their ability to send data via a Wi-Fi network. They consist in two pieces containing magnets: in case of an opening, the change of the magnet position causes the electric circuit to close. An opening state information is then sent. Another state information is sent in case of closing. The data is collected using the same Wi-Fi network as the one used for the indoor ambiance measurement and is stored on the

university's servers¹².

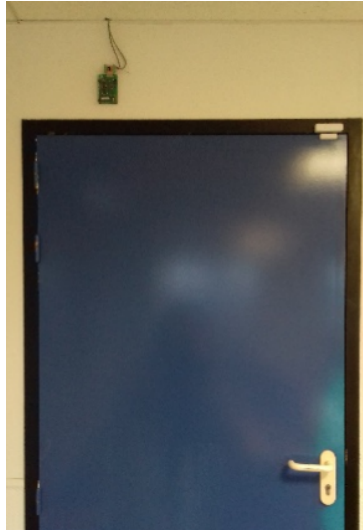


Figure 2.13 – A sensor placed on a door (upper right corner).

All windows facing the outside have been equipped with sensors in February 2022. The doors to the corridors were equipped in September 2022. No sensors were installed on the windows to the corridors, nor on the doors to the adjacent rooms, which are only supposed to be opened in case of emergency. The locations of sensors and their identification in the measurement database are shown in Figure 2.14.

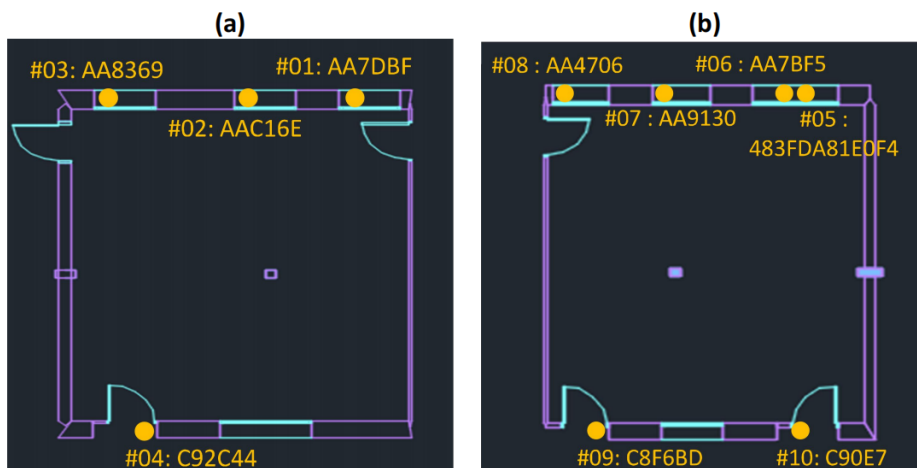


Figure 2.14 – Names and locations of windows and doors sensors in room 114 (a) and 219 (b).

12. <https://biot.u-angers.fr/shelly.php>

2.3.3.3.2 Electricity consumption The electricity consumption of the monitored rooms is followed to identify various actions, such as the use of light, computers or projectors. The Gulplug Company Save-it-yourself solution was chosen to record the instantaneous power of the different equipment. It consists of a magnetic core (Figure 2.15 a) placed around the wires of each electrical appliance to be monitored. The measured power is sent wirelessly to a box (E-Access on Figure 2.15 b), and transmitted to a datalogger (E-Log on Figure 2.15 b) capable of sending data to a server using the GSM network. The students do not have access to the electricity consumption monitoring equipment.

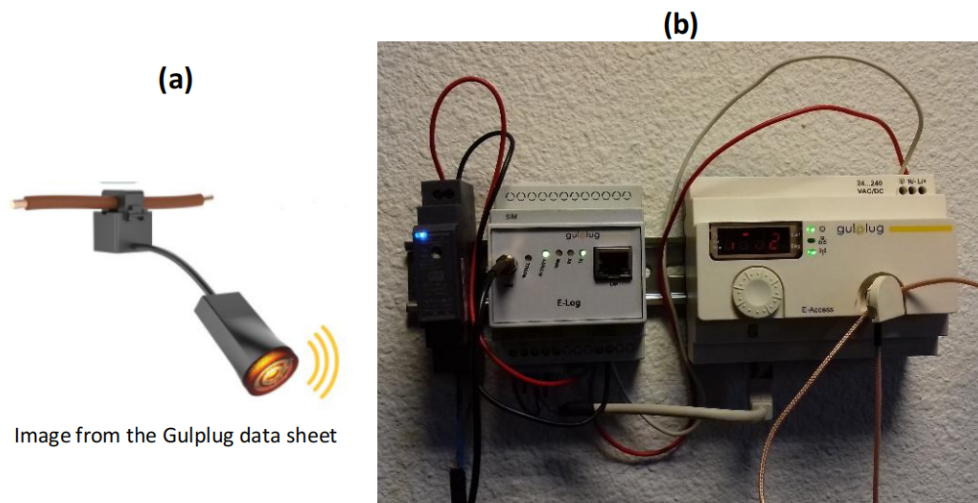


Figure 2.15 – Magnetic core (a) and recording device (b) to monitor the electricity consumption.

The data has been recorded every minutes since February 2022 and is accessible online via the equipment manufacturer’s platform¹³. Overall electricity and lighting consumptions are monitored in both rooms. In room 114, the consumption of the sockets and video projector are also measured.

Table 2.6 – Measurement deployment in rooms 114 and 219

	Room 114	Room 219
Global electricity consumption	1 core	3 cores (one per phase)
Lighting	1 core	1 core
Sockets	1 core	-
Video projector	1 core	-

13. <https://www.save-it-yourself.com/index.php>

2.3.3.3.3 Occupants presence The different sensors presented in the previous paragraphs give information on the use of the building: actions on the windows or on the equipment. A priori, these actions are performed when occupants are present in the room. However, occupants can be present without interacting with windows and systems. Furthermore, occupants may leave the room without switching off electrical devices. Therefore, an alternative strategy was proposed to obtain labelled data on the presence of occupants. Instead of installing a presence sensor capable of counting the number of entries and exists in the rooms (a method that requires additional sensors and has proven to be unreliable in other experiments), it was decided to rely on the use of a so called “soft-sensor” in this study. Here, school’s time management tool will be used to find out if and when the rooms were occupied. No additional sensors are installed. This strategy also has some drawbacks:

- Only the presence of occupants for a class is reported, which excludes the presence of the cleaning staff in the morning, as well as the presence of students or teachers for meetings during the day.
- Teachers may decide to teach in another room at the last minute, without notifying it in the tool, i.e. the room may appear as booked while it was empty (and the reverse situation is also possible).
- Only the teaching slots appear as booked in the tool: when a group stays in a room for two consecutive slots, there is uncertainty about the presence of the group during the breaks.

Room occupancy can be exported from the school’s time management tool and the data are stored on the university’s servers. Additionally, a paper survey is available in the two monitored rooms. On a voluntary basis, occupants can report their presence, as shown in Figure 2.16. This data is manually added to the database. A paper version, placed near the entrance door, seemed more appropriate than a numerical version for this additional occupancy reporting: we considered that users would be more likely to report their presence if they did not need to log in to an application. The paper survey is complemented by a project explanation sheet.

2.3.4 Data recorded

It is not always possible to process the recorded raw data as we will see in the following chapters. Gaps and outliers have been observed in the recorded raw data. A first pre-processing is therefore required. In the framework of the BIoT project, the pre-processing



Figure 2.16 – Survey and information on the monitoring available in the rooms.

of the data was investigated together with a master student: Ahmed Es-Sabar. The first pre-processing consisted of finding outliers. A measured value was identified as an outlier if it was outside the measurement range of the corresponding sensor. In this case, the measurement point was removed from the database. The second pre-processing was the treatment of missing data that could occur due to technical problems with a sensor or the data collection devices. The process of replacing missing data with probable data is called data imputation and has been studied by Es-Sabar (Es-Sabar et al. 2022; Es-Sabar 2022). Many imputation methods are available in the literature (Hasan et al. 2021; Weerakody et al. 2021; Es-Sabar et al. 2022). Some methods simply remove the features for which data are missing or replace the missing value with a constant value (i.e., zeros, mean, last or next observed values). These methods can introduce a bias into the data processing (Luo et al. 2018; Osman et al. 2018) and advanced imputation approaches based on machine learning (Chong et al. 2016; Pazhoohesh et al. 2019; Cho et al. 2020) and deep learning (Che et al. 2018; Fouladgar et Främling 2020; Okafor et Delaney 2021) have recently been proposed. In the BIoT project, the choice of the imputation method has been discussed in order to identify the method that performs the best regressions for estimating the electricity consumption of the classroom. In the developed methodology, 10 to 60% of data is first removed from the database using different missing data generation mechanisms. Then, a set of ten imputation methods was tested. The performance of the imputation

was evaluated by comparison with the original complete data and quantified using the RMSE indicator. As an example, the imputation performance of the ten methods for CO₂ measurements are given in Figure 2.17 from (Es-Sabar 2022). The Missing Completely at Random¹⁴ (MCAR) missing data generation mechanisms were used in this example in order to remove 10 and 60% of the available measured data. It turns that the linear, polynomial, and quadratic interpolations (LI, PI and QI) were the best performing imputation methods regardless of the features to input (e.g. CO₂, temperature, . . .), of the amount of missing data, and of the missing data generation mechanism (Es-Sabar 2022).



Figure 2.17 – Comparison of the RMSE for some CO₂ sensors of room 219 using the MCAR missing data generation mechanism and for 10 % (a) and 60 % (b) of missing data. From (Es-Sabar 2022).

The effect of the imputation method on the final task (prediction of electric power in room 219) is shown in Figure 2.18. As in the previous example, the MCAR missing data generation mechanisms were used in order to remove 10 and 60% of data. It turns out that the performance is insensitive to the imputation method: regardless of the imputation method used, the performance of the final task remains similar (Es-Sabar 2022). These results were confirmed by another case study (Es-Sabar et al. 2022). Furthermore, it was found that the feature extraction steps are more important than the choice of an imputation method in improving the performance of the final task (Es-Sabar 2022).

In this thesis, the methodology developed in the BIoT project was used and the following pre-processing was performed. Outlier values were filtered out following the above-mentioned procedure. Then, when data is missing, two cases are considered. Firstly, if data is missing more than 30% of the time for a feature, this feature is not further considered in the assessment. Secondly, for shorter data gaps, the linear interpolation method

¹⁴. In the MCAR mechanism, the distribution of missing data for one input is independent of the distribution of missing data for others inputs, and it is also independent of the missing value itself.

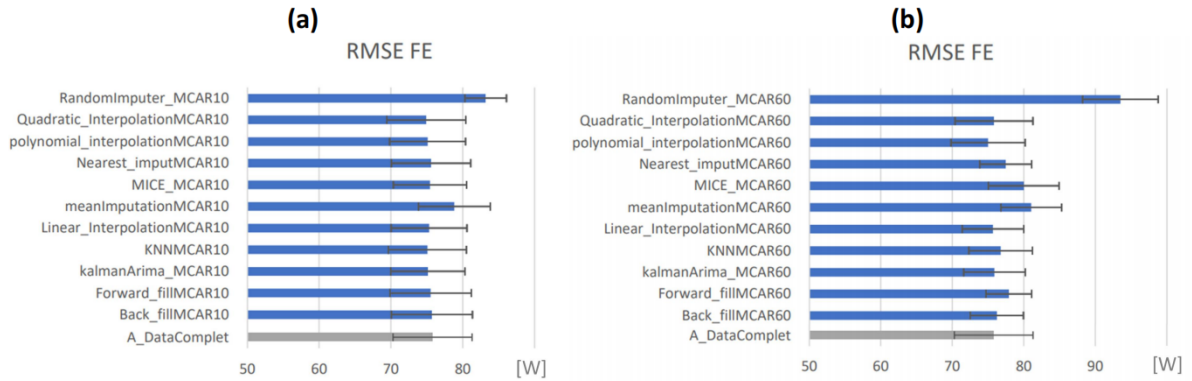


Figure 2.18 – Comparison of the RMSE on the final regression task using the MCAR missing data generation mechanism for 10 % (a) and 60 % (b) of missing data. Adapted from (Es-Sabar 2022).

was chosen to capture the data. If data is missing at the beginning of the time serie, it is replaced by the first known value. Conversely, if data is missing at the end of the time serie, it is replaced by the last known value. The measurements obtained after the pre-processing are available in Figure 2.19 to Figure 2.28. The graphs are given for the last seven days of September 2022 (from Saturday September 24 to Friday September 30). The indoor measurements in the graphs were performed in room 219, by sensors close to the windows.

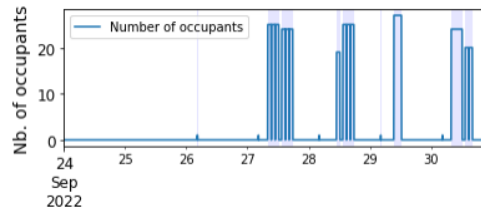


Figure 2.19 – Number of occupants

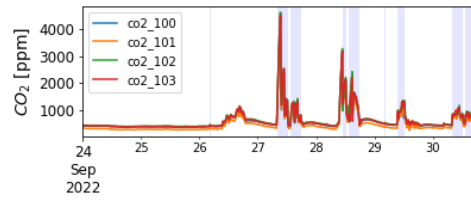


Figure 2.20 – CO₂ concentration

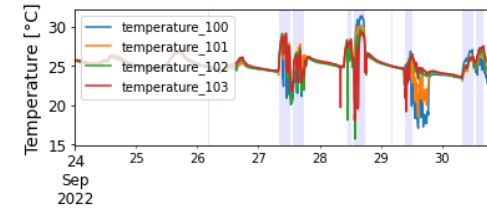


Figure 2.21 – Indoor temperature

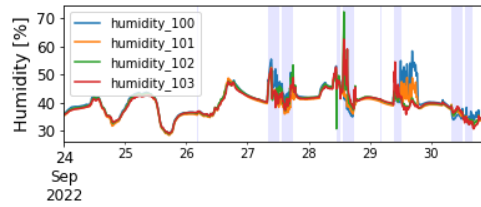


Figure 2.22 – Humidity

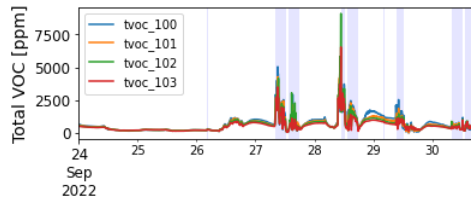


Figure 2.23 – Total VOC

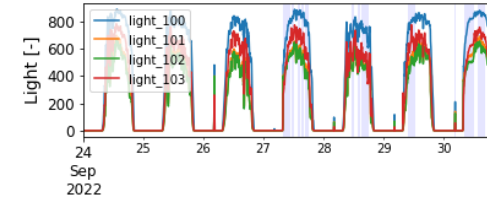


Figure 2.24 – Light

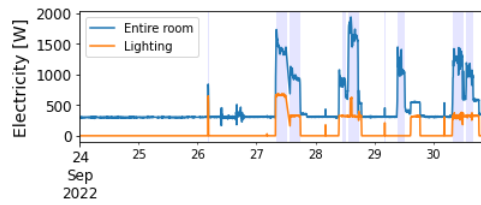


Figure 2.25 – Electricity consumption

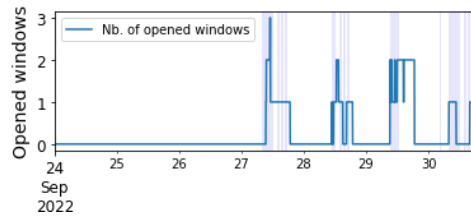


Figure 2.26 – Opened windows

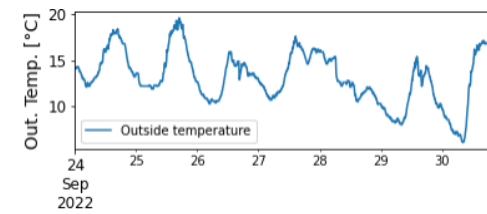


Figure 2.27 – Outdoor temperature

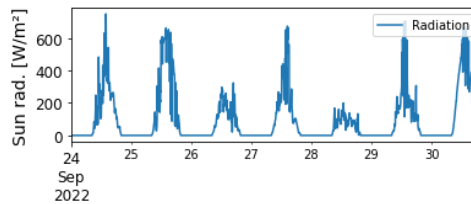


Figure 2.28 – Sun radiation

Correlations are clearly visible between the presence of occupants (Figure 2.19) and other features such as: CO₂ concentration (Figure 2.20), total VOC (Figure 2.23) and electricity consumption (Figure 2.25). In addition, when occupancy is present, temperature (Figure 2.21) and humidity (Figure 2.22) are more likely to vary, which is related to the openings of windows (Figure 2.26). On Monday September 26, for example, the CO₂ concentration (Figure 2.20) and total VOC values (Figure 2.23) suggest that a small group of occupants was present, but this was not reported on either the time management tool or on the paper presence survey (no corresponding lightblue background). In September, the brightness feature (Figure 2.24) is not very representative of the occupancy: the occupants are mainly present during the day, and the brightness value follows the solar radiation values (Figure 2.28). Only the presence of cleaning staff is visible in the early morning. In case of absence, all multiphysical sensor boards of a room give consistent measurements. A significant deviation between the boards is only visible in case of presence. For instance, the CO₂ concentration measured by a sensor is higher if occupants are close to this sensor, or lower if the sensor is placed near an opened window. In view of this first analysis of the data measured data Polytech Angers, the actions of the occupants should be trackable with the algorithms chosen in the following chapters.

2.4 Conclusion of the chapter

The two case studies that will be use throughout the thesis have been presented in this chapter. For the first case study, DBES and CFD simulations of a virtual educational building are performed using DesignBuilder. The software and the modelling assumptions have been described. The simulations are performed for different occupancy scenarios in order to model a variety of uses and to have enough data to train the algorithms applied in the following chapters. As an output, the temperature evolution is obtained throughout the simulated period for a thermal zone or at different location within a thermal zone. The inputs and outputs of the DBES will be extracted from the software and used in chapters 4 and 5, to train machine learning algorithms to identify presence of occupants and their actions on the windows. In addition, the CFD results will be used in chapter 3 to select the set of sensors that best describe the user's actions and comfort conditions. In the second case study, real data from two classrooms of Polytech Angers are recorded. The data collection strategy has been developed in the framework of the BIoT project, in which this thesis is part of. The monitoring has been fully described in this chapter. It

consists of multiphysical sensors designed and produced at Polytech Angers to monitor the indoor ambiance. The outdoor conditions are monitored using a weather station. Finally, occupancy is detected using window opening sensors, smart-meters and the room occupancy schedule. All measured data are used for activity detection purpose in chapters 4 and 5, to study the optimal placement of sensors in chapter 3.

OPTIMAL SENSOR LOCATION

This chapter focuses on the problem of optimal sensor location. To this end, a comprehensive review of the available methods and their importance in various fields is first proposed. After evaluating multiple techniques, the Effective Independence Method (EIM) and Information Entropy (IE) methods emerged as the most promising solutions. Both methods were explained in detail. Their efficacy was tested through simulations that mirrored real-world deployment scenarios. The optimality of sensor placements was gauged using model stability and Root Mean Square Error (RMSE) metrics. In complex environments, we explored multi-zone and multi-wall scenarios, testing the flexibility of the chosen methods. Further validation was achieved by applying these techniques to real-world data, assessing predictions and RMSE. While our findings are optimistic, it is paramount to note that the term 'optimal' is contingent on specific application contexts. Future endeavors should aim to refine these techniques, promoting their adaptability to diverse conditions.

3.1 Introduction

As mentioned in Chapter 1, finding the optimal location for sensors is a critical aspect of various scientific, engineering, and industrial fields. It encompasses a wide range of applications, including structural health monitoring (Kammer et al.1991), environmental monitoring (Dong et al.2018), and acoustic control, geological disaster detection (Padula et al.1998, Lee et al.2018). Sensor location is a crucial aspect in our study enabling an accurate occupant behavior detection and efficient user-centric predictive control. The purpose of occupant behavior detection in a smart building is to monitor and analyze the activities and behaviors of people inside the building in order to improve their comfort, security, and the building's energy efficiency. The location of the sensors monitoring the activities can greatly affect the quality of the data collected and hence the accuracy of the machine learning models. The optimal placement of these sensors is important to capture the spatial variability of environmental parameters (like temperature and humidity) and

the dynamic effects of the building’s occupants and systems. Poor sensor placement can lead to inadequate or misleading data, which can cause the user centric predictive control to make suboptimal or even incorrect control decisions, leading to discomfort for the occupants or unnecessary energy use.

However, determining the optimal sensor location is a challenging task that involves trade-offs between conflicting objectives, such as data accuracy, the number of required sensors, and the cost of sensor deployment. In many applications, the goal is to minimize data uncertainty while maximizing the accuracy of the results. This necessitates careful consideration of the measurement process, the environment, and the desired objectives.

In this Chapter, we will first present in section 3.2 two selected methods for finding the optimal sensor locations, addressing the specific cases of this thesis. Secondly, in section 3.3, we will apply these methods to our two study cases: simulation and real scenarios, as mentioned in Chapter 2. The problem of optimization will then be formulated and also discussed in detailed terms. Finally, we will analyze and evaluate the result in section 3.4, 3.5 and 3.6.

3.2 Complementary state of the art and theoretical aspects

In the next subsections and regarding the conclusions of the state-of-the-art in Chapter 1, we will explore further two selected methods for determining the Optimal Sensor Placement (OSP) for temperature measurement.

The first method, known as the Effective Independence Method (EIM), proposed by Kammer (1991), involves analytical calculations for certain aspects but primarily relies on numerical approaches to determine the optimal sensor positions. As a result, this method inherits the advantages of analytical methods, such as computational speed, as it employs closed-form equations or formulas. Furthermore, the EIM method also exhibits some advantages of numerical methods, including accuracy, as it relies on numerical computation techniques that can deliver more precise and robust solutions. Moreover, it demonstrates adaptability by effectively handling complex and realistic scenarios using detailed models, rendering it suitable for a wide range of OSP problems. Kammer. (1991) argued that the optimal arrangement for measuring and estimating structural vibration can be achieved by minimizing the norm of the Fisher information matrix (FIM) constructed from modal and measurement covariance matrices. EIM aims to maximize sensor efficiency in terms

of spatial coverage and measurement accuracy, identifying optimal positions that provide maximum system coverage while minimizing the total number of sensors required. This approach yields precise measurements with fewer sensors, reducing installation and maintenance costs. Additionally, EIM considers the noise term in the linear model, making it more robust to noise by incorporating the noise variance when computing the Fisher information matrix. Another strength of the EIM is its ability to optimize multiple objectives simultaneously, allowing for the consideration of various performance criteria in sensor placement decisions.

The second method we will explore is the information entropy (IE) method, which leverages the concept of entropy as a measure of uncertainty in a probability distribution. IE captures the diverse information provided by sensors, making it an advantageous approach for sensor placement. By maximizing the IE, the sensor placement problem aims to maximize the diversity of information captured by the sensors, resulting in a more informative system. This method is based on the probability distribution of sensor measurements and can effectively guide the selection of sensor locations.

Given the strengths and advantages of the EIM and IE methods, both are well-suited for the task of finding optimal sensor locations for temperature measurement in indoor environments. In the following subsections 3.2.1 and 3.2.2, we will provide detailed explanations of these two methods: Section 3.2.1 will focus on the EIM, presenting its theoretical elements and computational techniques, while Section 3.2.2 will deal with the IE method, discussing its underlying principles and practical applications. Subsequently, in section 3.3, we will adapt these methods to our specific context of temperature spatial measurement, estimation, apply them to simulation and real case studies to compare their performance and effectiveness.

3.2.1 The effective independent method (EIM)

The EIM is widely used in the field of mathematical physics and has demonstrated success in OSP for structural health monitoring (SHM). This method, initially proposed by Kammer (1991) involves discretizing the studied system, considering a virtual sensor at each point of the mesh. The information matrix is then formed using a modal matrix and the points are sorted based on their contribution to the independence of the target modal matrix (which can be seen in Equation 3.1). The rank of this matrix, denoted as $\text{rank}(\Phi)$, is a measure of the linear independence of its column vectors (the mode shapes in the original work of Kammer). It is related to the maximum number of linearly independent

column vectors in the matrix.

In the original Kammer's study, a large structure for spatial application was studied. This structure was represented as a finite element model (FEM), which discretizes the geometry into interconnected nodes representing the elements of the structure (e.g., beams, plates). These nodes define the degrees of freedom (DOFs) of the structure, such as translational and rotational directions. By performing modal analysis using the FEM, the natural frequencies, mode shapes, and modal damping ratios of the structure can be determined. The mode shapes represent the displacement patterns of the structure at its natural frequencies, and the modal matrix Φ is assembled using these mode shapes. For a structure with N degrees of freedom (DOF) and r modes of interest, the modal matrix Φ is an $N \times r$ matrix, where each column represents a mode shape associated with a specific natural frequency. This modal matrix Φ is given by :

$$\Phi = [\Phi_1, \Phi_2, \dots, \Phi_r] \quad (3.1)$$

where Φ_i is the i -th (ranging from 1 to r) mode shape vector of the structure:

$$\Phi_i = [\phi_{1i} \cdots \phi_{ji} \cdots \phi_{Ni}]^T \quad (3.2)$$

where each ϕ_{ij} provides information about the relative displacements (or any other modal quantity, such as speed or acceleration) associated to the i -th mode and the j -th DOF (ranging from 1 to N).

To assess the contribution of each sensor to the independence of the mode shapes, an Effective Independence measure is derived from the determinant of the Fisher information matrix. The Fisher information matrix, represented by F , is formed as $F = \Phi^T \Phi$.

The independence measure is then defined as:

$$\text{Independence measure} = \det(\Phi^T \Phi) \quad (3.3)$$

By maximizing this measure, it aims to choose sensor locations that provide the most independent information about the mode shapes of the structure.

The number of virtual sensors and measurement points is then progressively reduced by eliminating points that contribute the least to the rank of the modal matrix. This involves optimizing the Fisher information array while preserving the linear independence

of the modal vector to the greatest extent possible. It is based on this principle that the optimal number and location of sensors will be determined.

An alternative effective independence measure E_D is calculated using the modal matrix (Φ), the eigenvalues (λ), and eigenvectors (φ) of the Fisher information matrix. The E_D coefficient represents the magnitude of the contribution of a given sensor position to the linear independence of the target modal matrix Φ :

$$E_D = \Phi\varphi \otimes \Phi\varphi\lambda^{-1} \quad (3.4)$$

Also, the ED can be calculated directly from the diagonal elements of the projection matrix formed by the modal matrix :

$$E_D = \text{diag} \left(\Phi \left| \Phi^T \Phi \right|^{-1} \Phi^T \right) \quad (3.5)$$

where $\text{diag}()$ represents the extraction of the diagonal elements of the matrix in the brackets, and the matrix $\Phi \left| \Phi^T \Phi \right|^{-1} \Phi^T$ is the projection matrix of the vector space tensed by the modal matrix Φ .

Another indicator derived from the Fisher information matrix is its "condition number". This is a measure of its numerical sensitivity. It quantifies the stability and reliability of calculations based on the matrix. Specifically, the condition number of a Fisher information matrix is defined as the ratio between the largest and smallest singular values of the matrix. It represents the ratio between the largest and smallest eigenvalues of the matrix. A high condition number indicates that the Fisher information matrix is ill-conditioned, meaning that small perturbations in the input data can result in large variations in the computed results. This can make numerical calculations unstable and the results less reliable. On the other hand, a low condition number indicates a well-conditioned Fisher information matrix, where calculations are less sensitive to numerical perturbations and more reliable. The condition number will be used to guide the selection of rows to remove from the Fisher information matrix. The row whose removal leads to the smallest condition number is selected, which means that removing this row has the least impact on the overall information contained in the matrix, making it the most "independent".

To summarize, the EIM can be divided into the following iterative steps:

- **Step 1: Identification of the least contributing sensor:** Find the sensor position corresponding to the smallest element in the EI measure. This position represents the sensor that contributes the least to the independence of the target

modal matrix Φ .

- **Step 2: Deletion of the least contributing sensor:** Eliminate the row corresponding to this sensor from the Fisher information matrix. This step effectively removes the least contributing sensor from consideration in subsequent iterations.
- **Step 3: Reconstitution of the Fisher Information Matrix:** With the row corresponding to the least contributing sensor removed, recalculate the Fisher Information Matrix.
- **Step 4: Repetition of the Process:** Repeat steps 1-3 iteratively, each time identifying and removing the sensor that currently contributes the least to the independence of the target modal matrix Φ .

The iteration continues until the desired number of sensors is reached. This approach ensures that the sensors that remain are those that contribute the most to the independence of the target modal matrix, effectively optimizing sensor placement.

3.2.2 The information entropy (IE) method

The IE method is a powerful approach in optimal sensor location, based on the concept of information theory, introduced by Claude Shannon (1948). Information theory provides a mathematical framework for quantifying and analyzing the amount of information conveyed in signals or data. A key concept in information theory is information entropy, which measures the uncertainty or randomness in a system. In the context of OSP, the objective is to find sensor locations that provide the most informative and diverse measurements about the system.

The IE method aims to achieve this by selecting sensor locations that maximize the information entropy, which quantifies the average amount of information required to describe the outcome of a random variable.

Mathematically, the information entropy H for a discrete random variable X with probability mass function $p(x)$ can be defined as:

$$H(X) = - \sum_{x \in X} p(x) \log p(x) \quad (3.6)$$

Maximizing the information entropy or, equivalently, minimizing the uncertainty in the system is typically achieved by solving an optimization problem where the sensor locations are the decision variables, and the information entropy is the objective function.

There are several approaches to implement the information entropy method in optimal

sensor location, including:

- Maximizing the mutual information between the sensor measurements and the underlying state of the system.
- Minimizing the conditional entropy of the system state given the sensor measurements.
- Employing greedy algorithms, heuristic approaches that make locally optimal choices at each step in order to achieve a globally optimal solution, that iteratively select the sensor location that provides the largest information gain at each step.

Lee et al. (2008) addressed the use of the IE method to determine the optimal location for a set of combination temperature sensors in a greenhouse. They proposed the following function to obtain the combined optimal sensor locations, aiming to minimize redundant information and maximize the amount of information for the combined sensors.

$$\sum_{i=1}^n T(X_i, X_j, \dots X_p) = H(X_k) + \dots + H(X_p) + \sum_{i=1}^{n-p} \sum_{j=k}^p H(X_i, X_j) \quad (3.7)$$

Here, n is the total number of sensors inside the greenhouse, p is the number of sensors that should be selected, $\sum_{i=1}^n T(X_i, X_j, \dots X_p)$ is the total information entropy.

The IE for sensor placement, inspired by the concepts of entropy in information theory, can be described as follows:

- **Step 1: Identification of Sensor Information Entropy:** Compute the entropy of the information provided by each sensor. The entropy is a measure of the uncertainty or randomness of the information.
- **Step 2: Ranking of Sensors:** Rank the sensors based on their information entropy. The sensors with higher entropy are considered to provide more diverse or uncertain information.
- **Step 3: Selection of Sensors:** Select the desired number of sensors starting from the one with the highest entropy. This ensures that the selected sensors provide the most diverse set of information.
- **Step 4: Iteration of the Process:** If necessary, repeat the steps iteratively until the entropy condition is satisfied or the desired number of sensors is reached.

This method assumes that sensors providing more diverse information (higher entropy) contribute more to the understanding of the system dynamics and thus should be preferred for placement.

3.3 Necessary adaptations before application on simulation data with EIM and IE method

As a recall of the simulation case discussed in Chapter 2, the simulated building, depicted in the Figure 3.1, is oriented towards the north and has dimensions of 10 m width, 12 m length, and 3.5 m height. On the left wall, the temperature values at $m=528$ points, representing the temperature measured by potential / candidate sensors, are evaluated through CFD simulations using the DesignBuilder software. In our first application of OSP methods, we started from simplest scenario by dealing only with one wall, in other words, sensors can only be installed on one wall which is the left wall in this figure. The optimization problem is to find the ($M =$) 3 optimal sensor locations among the ($m =$) 528 candidate locations able to provide the best estimation of the temperature at n target points. In this first application, we arbitrarily predefined $n=9$ target points taken at the same slice at 1 m-height. These 9 targets correspond to potential location of occupants in the room. The rationale for this approach is that we intend to choose the sensor on the wall to best describe their comfort.

Each CFD simulation represents the result for a single time point. Consequently, we performed $T=11$ separate CFD simulations (at different time intervals), spanning from December 1st at 7:00 AM to 5:00 PM. Each simulation accounts for updated boundary conditions.

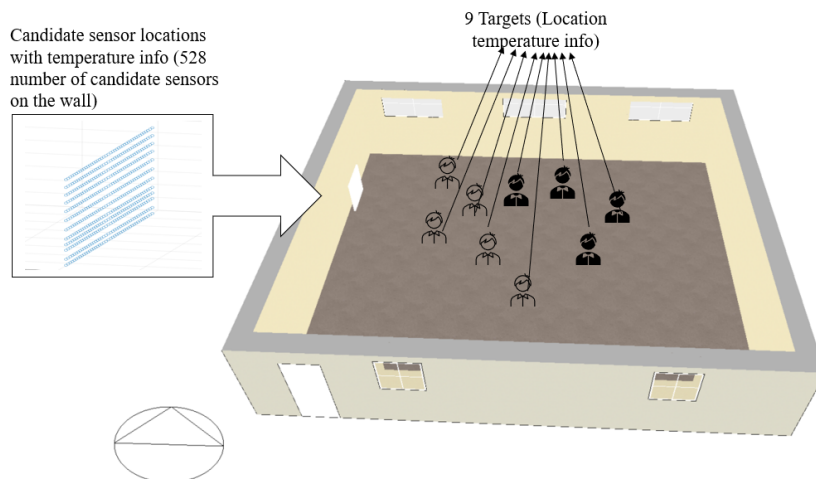


Figure 3.1 – Layout of the room and sensors ($m = 528$ candidate), targets ($n = 9$) locations

3.3.1 Adaptation and application of EIM

The EIM has been adapted to building performance context. Our goal is to selected the optimal set of sensors to monitor the indoor temperature at some specific or target location where occupants are supposed to be. The idea is to get information on their comfort without placing sensors exactly at people's location: sensors remain on the wall for convenience. An improved greedy algorithm, derived from the EIM, has been developed to identify the optimal sensor locations using simulation data from CFD.

The original EIM uses a measure of EI derived from the mode shape matrix of a structure, which essentially quantifies the extent to which a given sensor location provides information not captured by other locations. In our case, it calculates a measure of independence by computing the condition number of a matrix which results from a multiplication operation involving temperature readings from potential sensor locations and target locations. In addition, in the original EIM, after selecting a sensor location, the remaining rows of the mode shape matrix are re-orthogonalized to remain independent of the selected row. For our case, we exclude the row (i.e., the location) with the minimum condition number (which indicates maximum independence) from the matrix of potential sensor locations, we also applied the distance coefficient to avoid sensors cluster around the same points in the system with redundancy information. This algorithm leverages the condition number of the Fisher information matrix to enhance its performance.

In practical terms, the objective is to determine, through the iterative method presented at the end of section 3.2.1, the M optimal sensor locations among the m candidate locations on a wall to accurately and robustly infer the temperatures at n target (occupants) points located within the instrumented area of the room, based on the measurements collected at these M optimal locations. The key point is the determination of the Fisher information matrix.

Assuming the m potential sensors on the wall with T time intervals, and M represents the required number of optimal sensor locations (m representing all potential sensors, it should be much bigger than M), the U ($m \times T$) matrix, representing the temperature calculated or measured at the candidate sensor locations on the wall is expressed:

$$U = \begin{bmatrix} U_{11} & \dots & U_{1T} \\ \dots & \dots & \\ U_{m1} & \dots & U_{mT} \end{bmatrix} \quad (3.8)$$

Similarly, the q ($n \times T$) matrix, corresponding to the temperatures estimated at the target points inside the room, with the same T time intervals can be expressed:

$$q = \begin{bmatrix} q_{11} & \dots & q_{1T} \\ \dots & \dots & \dots \\ q_{n1} & \dots & q_{nT} \end{bmatrix} \quad (3.9)$$

A responses function φ , under the form of a $m \times n$ matrix, found in the initial EIM, representing the relationship between U and q matrices, i.e., between temperatures measured at the wall and temperatures inferred inside the room can be written as:

$$U = \varphi \cdot q = \sum_{i=1}^m \varphi_i \cdot q_i \quad (3.10)$$

The covariance matrix P , an unbiased estimator of the error between actual matrix q (i.e., actual temperatures at target points) and estimated \hat{q} (i.e., estimated temperatures through a least-square estimation of response function φ) can be given as:

$$P = E \left[(q - \hat{q})(q - \hat{q})^T \right] = \left[\frac{1}{\delta^2} \varphi^T \varphi \right]^{-1} = Q^{-1} \quad (3.11)$$

In the above equation, the δ represents the noise level or uncertainty associated with the measurements or observations. Q is the Fisher information matrix of dimension ($n \times n$), we look for in order to perform the EIM.

It is important here to open a parenthesis and provide some clarifications regarding the Fisher information matrix. Physically, the dimensions of a Fisher information matrix correspond to the number of parameters or variables that are being estimated or inferred in a given statistical model. Each element of the matrix represents the sensitivity of the model's likelihood function to changes in the parameters. The diagonal elements of the matrix represent the precision or information content associated with each individual parameter, while the off-diagonal elements capture the correlations or dependencies between different parameters. In the context of sensor placement, the dimensions of the Fisher information matrix would depend on the number of sensor locations being considered and the specific parameters being estimated or characterized by the sensor measurements.

For the sake of simplicity in the analysis, it is assumed that the calculated noise for each sensor is uncorrelated and possesses identical statistical properties.

Thus, the Fisher information matrix can also be re-written as:

$$Q = \frac{1}{\delta^2} \cdot A_0 = \frac{1}{\delta^2} \varphi^T \varphi \quad (3.12)$$

Since A_0 is equal to a multiplicative constant times the Fisher information matrix, it is this A_0 matrix that will serve as the basis for applying the EIM method. The OSP problem will therefore consist in identifying the rows (i.e., the sensor locations) of matrix A_0 that contributes the most to its stability or influences the least the covariance P of the estimation error, leading to the most robust or most unbiased estimation. We determine the target row in A_0 by employing the EIM algorithm, which focuses on finding the row with the smallest condition number, as calculated using the `cond()` function. Additionally, we introduce a distance coefficient. Initially, we establish a distance metric for potential sensor locations, which is based on the Euclidean distance in 3D sensor location space. These distances are then computed and recorded in a distance matrix, denoted as D , where D_{ij} represents the distance between sensor locations i and j .

Then, during each iteration of the main loop, where the condition number is calculated, we also calculate a penalty term for each potential sensor location. This is based on its distance to already selected locations.

$$\text{penalty} = \sum_{k=1}^n \frac{1}{D_{ik}} \quad \text{for } \text{priority}_k \neq 0 \quad (3.13)$$

The penalty term is then incorporated into the selection criterion. Instead of merely choosing the location with the smallest condition number, we select the location with the smallest adjusted condition number. The adjustment is a function of the penalty:

$$d_i = \text{cond}(F_i, 2) + \lambda \times \text{penalty} \quad (3.14)$$

Then in each iteration, the line with the minimum condition number combined with penalty (i.e., the best sensor location) will be deleted, and then an updated information matrix is calculated. This process is repeated one by one until the required number of sensors is achieved.

The pseudocode for optimal sensor location using adapted EIM is given in Algorithm 1.

Algorithm 1 EIM algorithm

```
1:  $m_j \leftarrow m$   $\triangleright$  %  $m$  is the total number of candidate locations (e.g., 528)
2: Initialize priority as an empty array
3: for  $j = 1$  to  $m$  do
4:   for  $i = 1$  to  $m_j$  do
5:     for  $k = 1$  to  $i - 1$  do
6:        $U_i[k, 1 : T] \leftarrow U[k, 1 : T]$   $\triangleright$  %  $T$  is the number of time intervals (e.g., 11)
7:     end for
8:     for  $k = i$  to  $m_j - 1$  do
9:        $U_i[k, 1 : T] \leftarrow U[k + 1, 1 : m]$ 
10:    end for
11:     $\delta_i \leftarrow U_i \cdot \text{pinv}(q)$   $\triangleright$  % Illustrative calculation of response function  $\delta$ , in
    Equation 3.11
12:     $A_{0_i} \leftarrow \delta_i^T \cdot \delta_i$   $\triangleright$  % Illustrative calculation of matrix  $A_0$  in Equation 3.12
13:     $d[i] \leftarrow \text{cond}(A_{0_i}, 2)$ 
14:  end for
15:  Find imin such that  $d[\textit{imin}]$  is the minimum value in  $d$ 
16:   $\textit{priority}[\textit{imin}] \leftarrow j/n$ 
17:  Update  $U$  by removing row imin
18:   $m_j \leftarrow m_j - 1$ 
19: end for
```

In real-world scenarios, the number of occupants and their distribution within a room can vary. Different occupancy levels can be represented as different scenarios, each with its own set of target points. To ensure that the selected sensor locations are optimal across various occupancy scenarios, we need to adapt the EIM algorithm to consider each scenario separately and then aggregate the results. For each occupancy scenario, the target matrix q will have a different set of rows, corresponding to the temperatures at the target points for that scenario. The response function φ and the subsequent matrices derived from it will also vary for each scenario.

The EIM algorithm can be adapted to handle multiple scenarios by introducing an outer loop that iterates over each occupancy scenario. For each scenario, the algorithm determines the optimal sensor locations based on the target points for that scenario. The results from all scenarios are then aggregated to select the final set of optimal sensor

locations.

3.3.2 Adaptation and application of IE method

Information entropy is a measure of the uncertainty associated with a set of data. In a similar scenario to EIM application, there are m potential sensors on a wall, and n target positions inside a room, each sensor measures temperature over T time intervals. The temperature measurements can be represented in the same matrix U of size $m \times T$:

To find the optimal sensor locations using IE, we need first to compute the $(m \times m)$ covariance matrix C for the sensor data matrix U .

$$C = \frac{1}{T-1} (U - \bar{U}) (U - \bar{U})^T \quad (3.15)$$

where \bar{U} is the mean of the sensor data matrix U .

The entropy H is a scalar value representing the uncertainty in the sensor data, and is derived from the covariance matrix C :

$$H = \frac{1}{2} \log((2\pi e)^m \det(C)) \quad (3.16)$$

where $\det(C)$ is the determinant of the covariance matrix C .

The final step consists in selecting the sensor locations that minimize the entropy H . This can be achieved by iteratively removing one sensor at a time and computing the entropy of the reduced covariance matrix. The sensor whose removal results in the smallest increase in entropy is considered the optimal sensor location.

In some cases, it may be desirable to find a set of sensors that collectively provide the best information about the target temperatures. To accomplish this, we can employ a combined optimization approach by considering multiple sensors simultaneously. One approach is to use a greedy algorithm that, iteratively adds sensor to the selected set in a way that maximizes the reduction in entropy at each step. The algorithm can be described as follows:

1. Determine the number of candidate sensor locations n and the number of target points m .
2. For each candidate sensor location, combine it with the target points and compute the entropy H of the combined system.

3. Repeat this process for all candidate sensor locations.
4. After calculating the entropy for each candidate location, identify and select the sensor locations that contribute to the maximum entropy.

The objective of this algorithm is to find the sensor locations that yield the highest information entropy, indicating they provide the most diverse and rich information about the system’s state. This approach offers a balance between computational complexity and solution quality. The pseudocode for the IE method can be found follows in the Algorithm (2) below:

Algorithm 2 Information Entropy algorithm

```

1:  $m \leftarrow$  number of rows in  $U$ 
2: Initialize  $entropy$  as a vector of zeros with length  $n$ 
3:  $cov\_U \leftarrow$  covariance of  $U^T$ 
4: for  $i = 1$  to  $n$  do
5:    $A \leftarrow$  concatenate  $U$  and  $q[i, :]$ 
6:    $cov\_A \leftarrow$  covariance of  $A^T$ 
7:    $entropy[i] \leftarrow 0.5 \cdot \log(\det(cov\_A) / \det(cov\_U))$ 
8: end for
9:  $max\_entropy\_index \leftarrow \operatorname{argmax}(entropy) \triangleright$  Find the index of the maximum entropy
10: Display  $U[max\_entropy\_index, :]$   $\triangleright$  Display the optimal sensor location
    corresponding to the maximum entropy

```

3.3.3 Further discussions about Optimal Sensor Placement

3.3.3.1 Discussion about information independence

The definition of optimal sensor location has traditionally focused on information independence which refers to the uniqueness of the information provided by each sensor in the system. In the context of temperature monitoring within building, it implies that each sensor offers distinct information about the temperature distribution in the room, which is neither redundant nor correlated with the information provided by other sensors. This is important because independent information enables a better overall understanding and control of the temperature distribution.

Mathematically speaking, information independence can be related to the rank of a matrix. This latter is the maximum number of linearly independent columns (or rows) it possesses. A higher rank indicates a system with more independent information. In the algorithm associated with EIM, maximizing the determinant of the information matrix A_0 of Equation 3.12 ensures that the columns of the matrix ϕ are as linearly independent as possible. This is because the determinant of a matrix is the product of its eigenvalues, and if any of the eigenvalues is zero, the determinant becomes zero, indicating linear dependence among the columns.

For example, consider an $n \times n$ square matrix Q , which has n eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$. We can decompose Q into its eigenvectors and eigenvalues using the following equation:

$$QV = V\Lambda \quad (3.17)$$

Where V is an $n \times n$ matrix composed of the eigenvectors of Q as its columns, and Λ is an $n \times n$ diagonal matrix containing the eigenvalues of Q :

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} \quad (3.18)$$

If the matrix Q is diagonalizable (i.e., it has linearly independent eigenvectors), we can write the following:

$$Q = V\Lambda V^{-1} \quad (3.19)$$

Now, we can compute the determinant of Q :

$$\det(Q) = \det(V\Lambda V^{-1}) \quad (3.20)$$

Using the properties of determinants, we can rewrite this as:

$$\det(Q) = \det(V) \det(\Lambda) \det(V^{-1}) \quad (3.21)$$

Since the determinant of the inverse of a matrix is the inverse of the determinant:

$$\det(Q) = \det(V) \det(\Lambda) \frac{1}{\det(V)} \quad (3.22)$$

The determinants of V cancel out:

$$\det(Q) = \det(\Lambda) \tag{3.23}$$

For a diagonal matrix, the determinant is the product of its diagonal elements, which are the eigenvalues of Q :

$$\det(Q) = \lambda_1 \cdot \lambda_2 \cdots \lambda_n \tag{3.24}$$

As you can see from this equation, if any of the eigenvalues λ_i is zero, the determinant of the matrix Q will be zero. By maximizing the determinant, we ensure that the columns of Q are as independent as possible, leading to a higher rank for the matrix. Maximizing the determinant of Q is therefore a key aspect of finding the optimal sensor location, as it guarantees the highest possible linear independence between the columns.

3.3.3.2 Discussion about stability

Stability, in the context of this problem, refers to the numerical stability of the system. A stable system is less sensitive to small perturbations or changes, such as variations in sensor readings due to noise or other factors. A stable system can provide more reliable and accurate measurements. Mathematically speaking, as already stated, stability can be related to the condition number of a matrix. A lower condition number indicates that the system is more stable and less sensitive to change. Specifically, the condition number is defined as the ratio of the largest eigenvalue to the smallest eigenvalue of the matrix. For example, consider a linear system:

$$Q\mathbf{x} = \mathbf{b} \tag{3.25}$$

where Q is an $n \times n$ matrix, \mathbf{x} is an $n \times 1$ vector of unknowns, and \mathbf{b} is an $n \times 1$ vector.

Now, let us consider a small perturbation in \mathbf{b} , denoted as $\Delta\mathbf{b}$. We want to find the perturbation in the solution \mathbf{x} , denoted as $\Delta\mathbf{x}$. And the perturbed system can be written as:

$$Q(\mathbf{x} + \Delta\mathbf{x}) = \mathbf{b} + \Delta\mathbf{b} \tag{3.26}$$

Expanding this equation and subtracting $Q\mathbf{x} = \mathbf{b}$ from both sides, we get:

$$Q\Delta\mathbf{x} = \Delta\mathbf{b} \quad (3.27)$$

Now, we want to find a bound on the relative change in \mathbf{x} due to the perturbation in \mathbf{b} . The relative change in \mathbf{x} can be represented as:

$$\frac{|\Delta\mathbf{x}|}{|\mathbf{x}|} \quad (3.28)$$

From the equation $Q\Delta\mathbf{x} = \Delta\mathbf{b}$, we have:

$$|Q\Delta\mathbf{x}| = |\Delta\mathbf{b}| \quad (3.29)$$

Now, we use the fact that the condition number ($\kappa(Q)$) of a matrix Q is defined as:

$$\kappa(Q) = |Q||Q^{-1}| \quad (3.30)$$

where $|Q|$ is the norm of the matrix Q . Using this definition, we can bound the relative change in \mathbf{x} :

$$\frac{|\Delta\mathbf{x}|}{|\mathbf{x}|} \leq \kappa(Q) \frac{|\Delta\mathbf{b}|}{|\mathbf{b}|} \quad (3.31)$$

This inequality demonstrates that the relative change in the solution \mathbf{x} is bounded by the condition number of Q multiplied by the relative change in \mathbf{b} . A larger condition number means that the system is more sensitive to perturbations in \mathbf{b} (noise), while a smaller condition number suggests less sensitivity to noise.

In the context of the EIM, the matrix Q represents the Fisher information matrix, and the condition number of this matrix measures the sensitivity of the system to noise. By maximizing the determinant of Q , the EIM indirectly reduces the condition number, leading to a more stable system at the optimal sensor locations. Additionally, the EIM incorporate the noise term in the linear model. When computing the Fisher information matrix Q , it accounts for the noise variance δ , as seen in the Equation 3.12. Hence, the method implicitly considers the uncertainty introduced by the noise when determining the OSP.

On the other hand, the IE method aims to find sensor locations that maximize the information content provided by the sensors. It is achieved by considering the system's uncertainty, represented by the covariance matrix. This covariance matrix measures the

relationship between variables in the system. A larger determinant of the covariance matrix indicates greater variation and less correlation between variables, which corresponds to higher information content. By maximizing the entropy of the system, the IE method selects sensor locations that provide the most independent and informative data. These sensor locations are considered optimal as they offer the greatest potential for understanding and controlling the temperature distribution in the room.

3.3.3.3 Discussion about modal linearity

In the context of a sensor location problem, both the EIM and the IE method operate under the assumption of model linearity. This linearity implies that each sensor's readings correspond linearly to the state of the system. For instance, if we are considering temperature sensors, this would mean that the temperature reading from each sensor is a linear function of the actual temperatures at different locations in the room. This can be mathematically represented as:

$$\mathbf{y} = A\mathbf{x} + \mathbf{w} \tag{3.32}$$

where:

- \mathbf{y} is the vector of sensor readings,
- A is the sensing matrix, where each row corresponds to a sensor and each column corresponds to a location in the room,
- \mathbf{x} is the vector of the actual temperatures at different locations in the room, and
- \mathbf{w} is the vector of noise terms.

If the relationship between the sensor readings and the actual state of the room is perfectly linear and known, and if the noise \mathbf{w} is minimal, then the sensor readings \mathbf{y} would provide an accurate representation of the room's state \mathbf{x} . However, in real-world scenarios, there are often uncertainties in the sensing matrix A , and the noise \mathbf{w} can be significant.

This implies that even if a sensor provides readings that seem optimal based on past data, it might not necessarily be the best location to capture the true state of the room. The sensor's readings might be influenced by its specific location, nearby heat sources, airflow patterns, or other factors that do not represent the overall room environment.

Therefore, when choosing the optimal sensor location, it is essential to consider not just the sensor readings themselves but also how well those readings reflect the broader

environment. This is where methods like the EIM and IE methods come into play, as they aim to identify sensor locations that provide the most informative and representative data about the environment.

3.4 First applications and results on a single room case study

In this new section, we will apply the two methods detailed previously to the simple case of a single-zone room, as shown in Figure 3.1 (a more complex study will be conducted for a case with multiple zones in the subsequent section).

As discussed before, in the EIM, the primary objective in the optimal sensor location problem is to determine the M (1 in here) optimal sensor locations among the m (528 in here) candidate locations on a wall. This is achieved by iteratively minimizing the condition number of the matrix A_{0_i} , which is proportional to the Fisher information matrix Q , i refers to a specific candidate location among locations. The objective function can be mathematically expressed as:

$$\min_i \text{cond}(A_{0_i}, 2) + \lambda \times \text{penalty} \quad (3.33)$$

where $\text{cond}(A_{0_i}, 2)$ is the condition number of the matrix A_{0_i} in the 2-norm, and the penalty term is based on the distance to already selected locations.

For the IE method, the primary objective is to select sensor locations M (1 in here) that maximize the information entropy among the m (528 in here) candidate. The objective function can be mathematically expressed as:

$$\max_{\mathbf{s}} H(\mathbf{s}) \quad (3.34)$$

where \mathbf{s} represents the selected sensor locations, and $H(\mathbf{s})$ is the entropy of the selected sensor locations.

As we have written in the previous sections, both chosen methods are suitable for identifying the optimal placement of a defined number of sensors. Their principles and formalisms also encompass the ability to estimate temperatures at the target locations. Thus, the two methods can be compared on several aspects: the selection of the best sensor locations (whether both methods yield the same optimal positions for the sensors),

their stability, and accuracy.

To get straight to the point, the two methods do not propose the same optimal location, as can be seen in Figure 3.2 depicting the 528 candidate positions on the left wall. The optimal location suggested by the EIM method is marked in red, while the one based on IE is marked in blue. There is a heater on the left wall, marked by a green square.

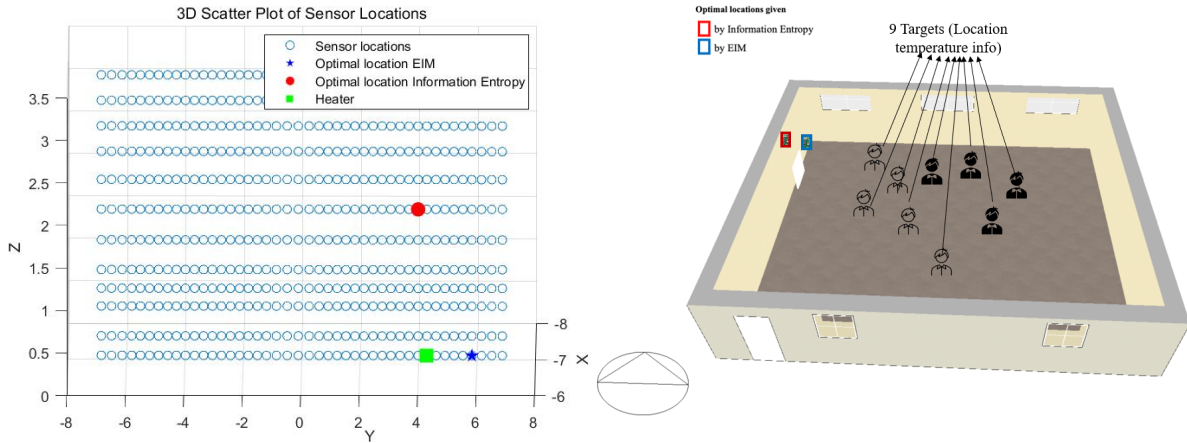


Figure 3.2 – Comparison of optimal sensor locations found from EIM and IE methods

As far as EIM is concerned, it selects the sensor location that minimizes the condition number of the resulting Fisher information matrix. The amount of data that each potential sensor location can collect regarding the temperature at the target locations is encapsulated in the Fisher information matrix (in the middle of the room). Since the heater is a primary source of heat, it causes significant temperature variation in its vicinity. The closer a sensor is to this heat source, the more sensitive it is to the changes in the system. It can capture finer details about temperature changes, making it a rich source of information. Additionally, in the EIM, a more stable and dependable system is implied by a lower condition number. Since they give less numerical instability and hence provide more precise estimations of the system’s status, sensor placements that produce lower condition numbers are thought to be preferable. Therefore, it makes sense that a location near the heater, which can provide ample information about the system’s state due to its proximity to the primary heat source, would result in a low condition number and be chosen as an optimal sensor location.

For the IE method, it focuses on maximizing uncertainty or entropy. It aims to choose sensor locations that provide the most uncertain or diverse set of data, effectively maximizing the potential information gain. In our context, the location close to the heater

might show a high degree of variability or uncertainty in temperature readings (due to the heater turning on and off, for instance), leading to a high entropy value and making it an attractive location for sensor placement.

Overall, it is not unexpected that both techniques selected areas near the heater. Placement of a sensor next to the heater would allow for close observation of the change in temperature, which would likely reveal important details about the room’s overall temperature dynamics.

In the first case study analyzed above, both the EIM and IE methods demonstrate their ability to determine the optimal location for a single sensor. Interestingly, these methods face no greater challenge when it comes to determine the placement of multiple sensors. Figure 3.3 illustrates the example of the definition of three optimal sensor locations for each algorithm are depicted. The IE algorithm determines locations that maximize information gain. Notably, the three red points, representing this algorithm’s optimal locations, are concentrated in the vicinity of the heater. This suggests that the algorithm recognizes the area near the heater as one of high informational value.

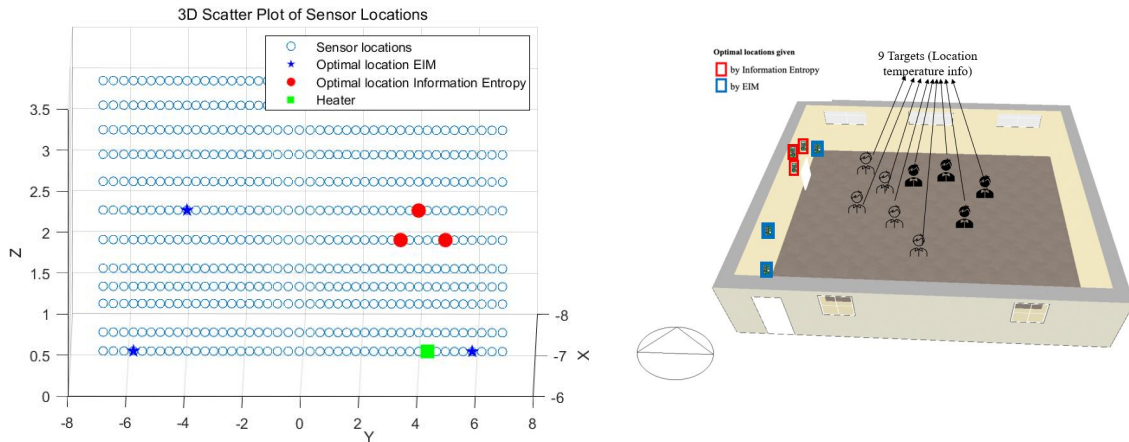


Figure 3.3 – Comparison of multiple optimal sensor locations from EIM and IE methods

On the other hand, the EIM algorithm pinpoints locations conducive to produce well-conditioned matrices, which are numerically stable and independent, facilitating subsequent computations. The primary objective of EIM is to minimize the correlation between data gathered by the sensors. Consequently, the optimal locations it selects, represented by the blue markers, tend to be distributed more widely across the room. This distribution ensures that the acquired measurements are as independent as feasible.

Given the distinct priorities of the two methods — one emphasizing information inde-

pendence and the other focusing on data uncertainty — it is reasonable to find disparities in the exact sensor locations chosen by each technique.

The next step is to verify and compare the two methods in terms of stability by assessing their sensitivity to measurement errors (generated by introducing noise around the reference values), and accuracy of the temperature estimations obtained from the sensors placed at their optimal locations.

3.4.1 Stability analysis

For the stability analysis, it is essential to ensure that the selection of optimal sensor location remains robust to various perturbations, including sensor errors, environmental changes, or other unmodeled factors. To assess this robustness, we simulate these perturbations by adding noise to original measured data.

We separate the process into four steps:

- **Add noise or perturbations to the data:** Modify the original data by introducing random noise or perturbations to simulate measurements uncertainties. We add a small random value to each data point using the equation below,

$$\mathbf{U}_{\text{noisy}} = \mathbf{U}_{\text{original}} + \epsilon \cdot \mathbf{A} \quad (3.35)$$

Here, $\mathbf{U}_{\text{noisy}}$ represents the noisy data, $\mathbf{U}_{\text{original}}$ is the original data, ϵ is the noise level, and \mathbf{A} is a matrix of random values with the same dimensions ($m \times T$) as the original data. The elements of these matrices are temperature values.

- **Re-calculate the optimal sensor locations:** Use the modified data to calculate the optimal sensor locations using the EIM and the IE method.
- **Assess the stability of the system:** Compare the optimal sensor locations obtained from the original data and the modified data. If the optimal sensor locations remain similar or identical, it indicates that the system is stable and robust against noise and perturbations. Conversely, significant variations in the optimal sensor locations suggest that the system may be sensitive to noise and perturbations.
- **Compare with other sensor configurations:** Perform the same stability analysis for other randomly sensor configurations which from the walls and compare their performance with the optimal sensor locations. The optimal sensor locations should demonstrate higher stability and robustness against noise and perturba-

tions compared to other sensor configurations. In the case of the IE method, we randomly select ten sensors to compare their information entropy with the original optimal sensor in both the original and modified data.

In Figure 3.4, we illustrate as an example the stability of the EIM method. The y-axis represents the absolute difference between the original data and the noisy data for each candidate sensor position (on the x-axis, representing the 528 possible positions). Notably, the EIM method consistently identifies the same location for the best sensor (at position 488 out of the 528 possible positions), which demonstrates its high mathematical stability and robustness.

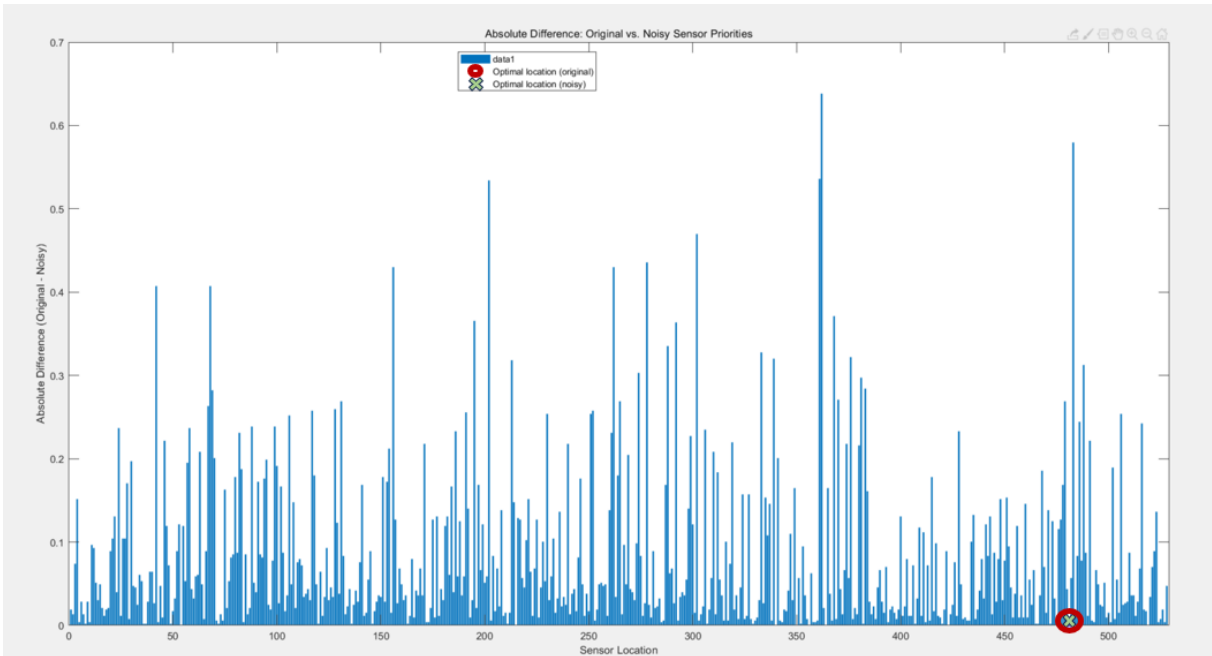


Figure 3.4 – Stability evaluation of the EIM

The results of the stability study of the IE method are presented in a different form in Figure 3.5 and 3.6. In Figure 3.5, the entropy values of the data are plotted on the y-axis for each of the 528 possible sensor positions. From this figure, we can observe that the optimal sensor location is not consistently selected in the same positions for the original data and modified data. However, the difference of information entropy when placing the sensors at their respective optimal location (for original and modified with noise or perturbations) is relatively small compared to those calculated for sensor locations taken randomly. This is reported in Figure 3.6, where the information entropy values are compared between the original and noisy datasets for the optimal location and other random location. The small

difference between the information entropy values for the original and modified data, with sensors placed at their respective optimal positions, supports the notion that this second method is also relatively stable and robust in the presence of noise and perturbations.

Overall, from the results, EIM is slightly more stable than the IE method, since the EIM inherently takes this uncertainty into account. Remember, one of the primary aims of the EIM is to ensure stability, which is closely tied to the ability to handle noise. This is because the EIM maximizes the determinant of the Fisher information matrix. When calculating this matrix, the method implicitly considers the noise variance. Hence, it is less sensitive to noise in the data.

In essence, the EIM focus on stability and its implicit consideration of noise make it more robust to noise in the data compared to the IE method.

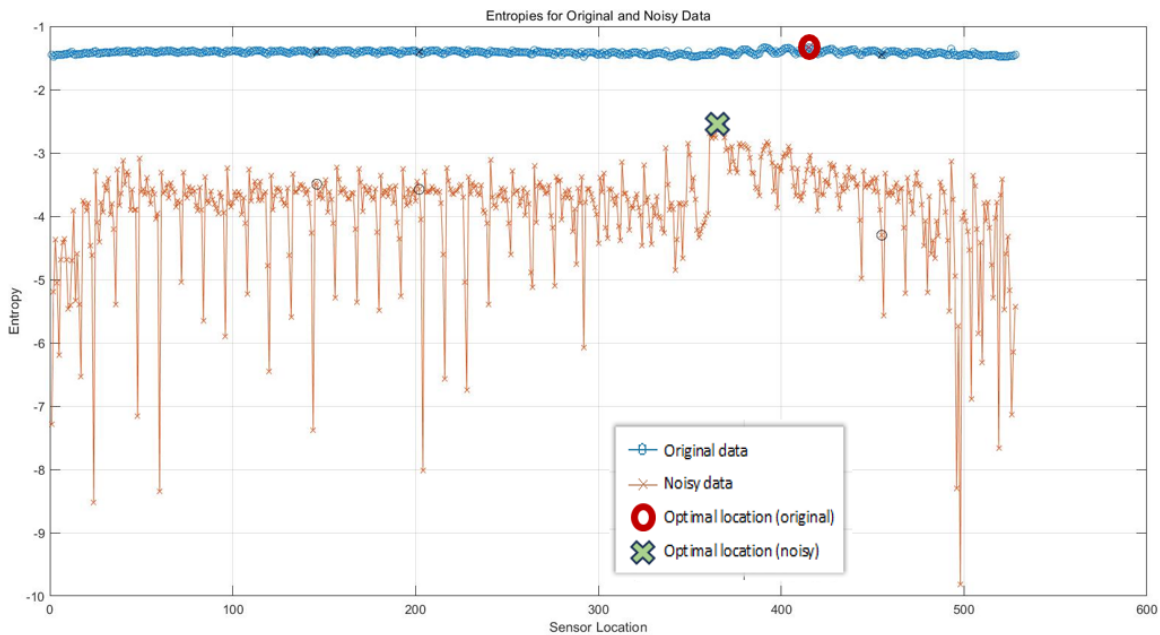


Figure 3.5 – Stability evaluation of IE method

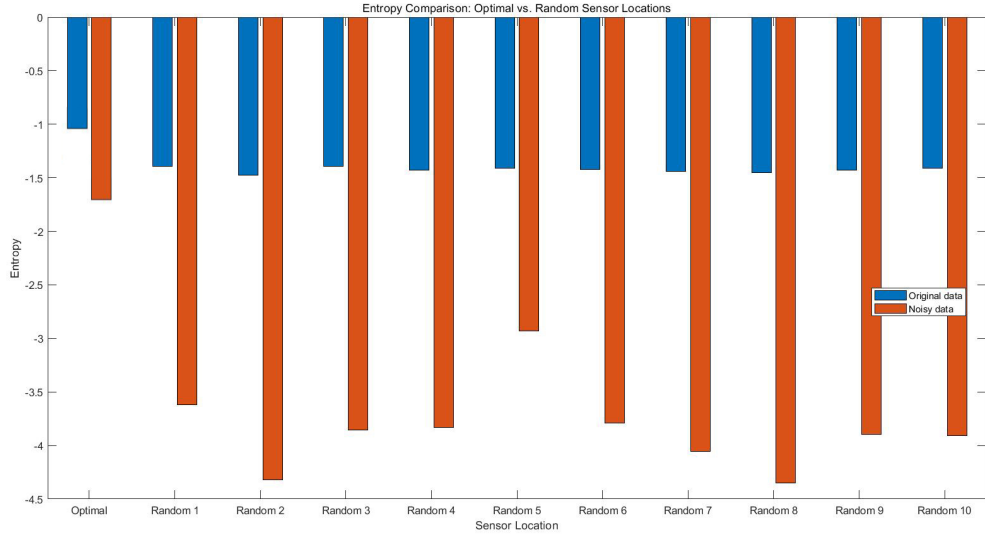


Figure 3.6 – information entropy for original and noisy data

3.4.2 Accuracy analysis

As mentioned in the introduction of section 3.4, we aim to evaluate the accuracy of the two methods when used to predict the temperature at the target points, thanks to their adapted formalisms. For that, we use the RMSE which is a common metric used to assess the performance of models or estimations by measuring the differences between the predicted values and the actual values. Considering that we have n target points, T time intervals, the RMSE is defined here as follows:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{1}{T} \sum_{j=1}^T (q_{ij} - \hat{q}_{ij})^2 \right)} \quad (3.36)$$

where q_{ij} and \hat{q}_{ij} represent respectively the actual (by CFD simulations) and predicted (using Equation 3.10) value of temperature at the i -th target point and j -th time interval.

Smaller RMSE values indicate better performance, as the differences between the predicted and actual values are smaller. Theoretically, the RMSE value obtained using the optimal arrangement of wall sensors should be the smallest or at least one of the smallest calculated values. To verify this, we randomly generated 1000 sets of triplets ($M = 3$ being the number of wall sensors we want to keep) representing wall sensor positions, to establish the response function ϕ of equation 3.10 for each of them, and

then calculated the estimated temperature values \hat{q}_{ij} at the $n = 9$ target points. Figures 3.7 and 3.8 below show the RMSE obtained for the 1000 random repetitions, and the specific RMSE obtained for the optimally defined location using the EIM and IE methods (indicated by the red line).

In an ideal scenario, as already mentioned, the red line (representing the optimal sensor location) should have a lower RMSE value compared to the majority of the random configurations. This would imply that the selected optimal sensor location provides better information than a randomly chosen configurations. From Figures 3.7 and 3.8, we observe that the red line is positioned to the left of the histogram (indicating a lower RMSE value compared to most random configurations). This result suggests that the optimal sensor location outperforms the majority of random configurations in terms of RMSE.

However, this also suggests that certain configurations may perform better than those obtained through our two methods. This arises from two factors with difficult-to-prioritize effects. The first contributing element to this situation, where configurations other than those provided by our chosen methods may be preferred, is the introduction of noise that disrupts the selection of optimal locations using our methods. The second factor to consider is the assumption of linear independence underlying both our approaches, which may not accurately reflect real-world conditions.

Nevertheless, we maintain a favorable view of our methods, especially the first method EIM, which provides better estimations than the majority of random configurations of wall-mounted sensors.

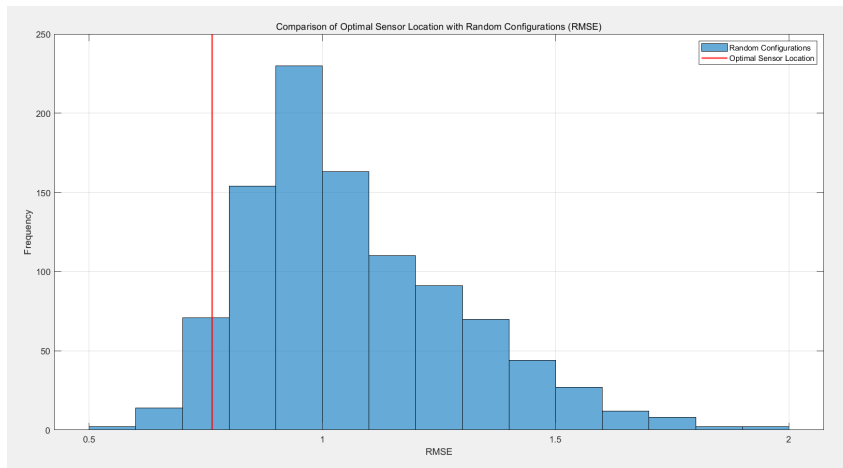


Figure 3.7 – RMSE of the value optimal sensor in EIM with target value

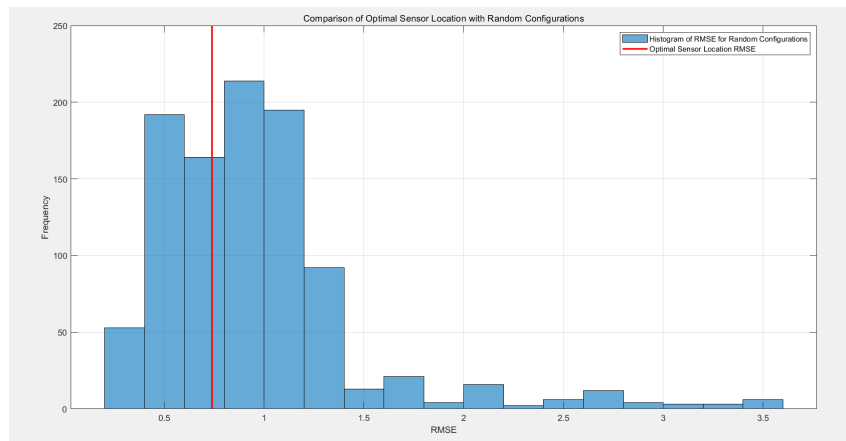


Figure 3.8 – RMSE of the value optimal sensor in IE methods with target value

3.5 Result for multi-wall and multi-zone

In this section, we will address two different scenarios. In both of them, the initial room size remains unchanged, and we still intend to find the optimal locations of $M = 3$ sensors among candidate locations. The first scenario (Figure 3.11) involves a multi-wall case where two walls in the room are considered as potential candidate locations for wall sensors, with 528 possible locations each. One of the walls (the left one) is situated close to a heater, resulting in a high-temperature distribution, while the other wall is farther the heater (no heater on this wall) and has a cooler temperature distribution. The $n = 9$ targets remain unchanged in this setup.

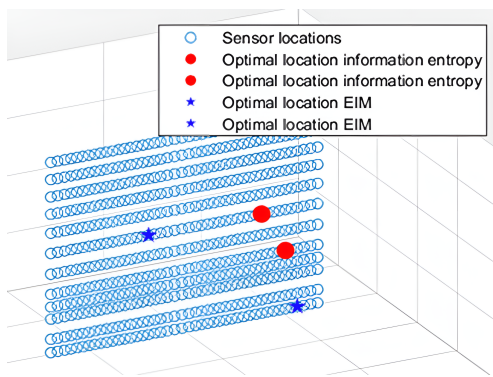


Figure 3.9 – At the left, a larger image of the point on Hot wall

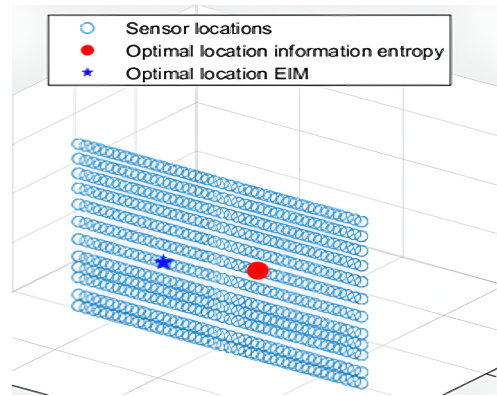


Figure 3.10 – At the right, a larger image of the point on Cold wall

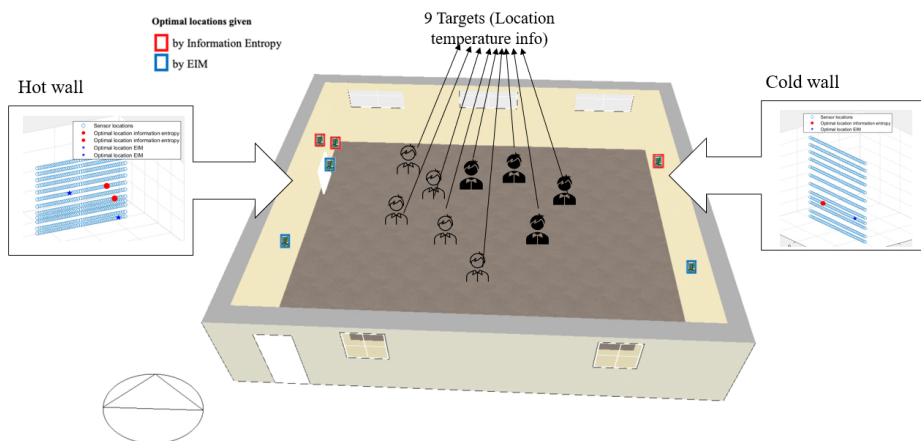


Figure 3.11 – Result for multi-wall configuration

The second scenario is a multi-zone configuration (Figure 3.14). The room is divided by a wall, and 9 targets are evenly distributed in the two zones. We select two walls that are farthest from each other as potential sensor locations, providing a total of 528 locations each.

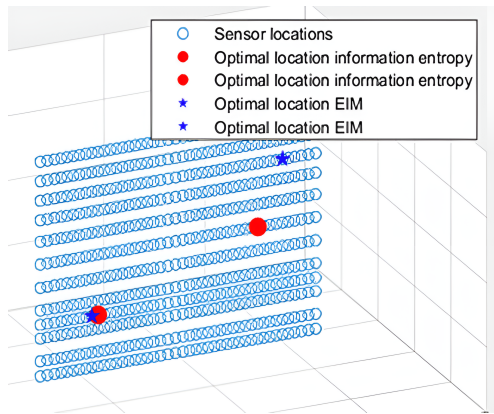


Figure 3.12 – At the left, a larger image of the point on zone left

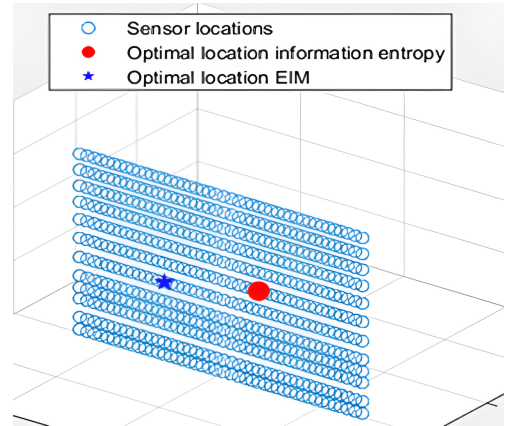


Figure 3.13 – At the right, a larger image of the point on zone right

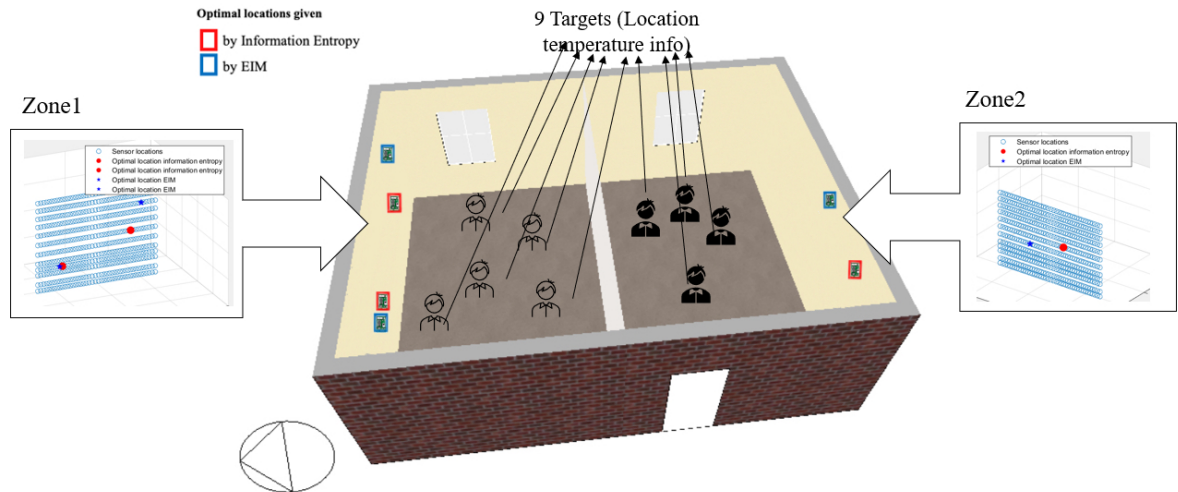


Figure 3.14 – Result for multi-zone

The observed results (in Figures 3.11 and 3.14) from both the EIM and IE methods underscore the difference in criteria when determining optimal sensor locations on the

walls. These distinctions arise from the inherent objectives of each method.

The EIM method prioritizes the independence of measurements. Its primary goal is to minimize the correlation between data collected by various sensors. Thus, the ideal placement of sensors, as determined by this method, tends to be dispersed across the room. This scattered distribution ensures that each sensor provides a unique perspective, minimizing overlap and redundancy in the data they collect. In Figure 3.11, the spread of the blue markers (representing the EIM method) on the warmer left wall is an evidence of this principle. This spread contrasts with the more clustered distribution of the red markers, which represent the IE method’s optimal locations. The latter’s concentration suggests a focus on specific areas, rather than a broad coverage.

The IE method is anchored in the principle of maximizing information extraction. The sensors’ ideal locations, as per this method, are those where temperature fluctuations are most pronounced. Such locations are information-rich, offering insights into the environmental dynamics. This principle is evident in the placement of the red markers near the heater on the left wall. The proximity of these sensors to the heater ensures that they capture the most significant temperature gradients, thus maximizing information gain.

A consistent observation for both methods is the recurrent preference for the left wall. In most scenarios, two of the three optimal sensors are positioned on this wall. This choice is logical, given that the left wall is most influenced by the heater, resulting in pronounced temperature variations. Such variations make this wall an information hotspot. On the other hand, the cooler right wall typically receives just one sensor. Though it may seem counterintuitive, this placement is strategic. By ensuring a sensor’s presence on the colder wall, the system guarantees a baseline measurement. This baseline aids in capturing the complete temperature spectrum, ensuring holistic monitoring while simultaneously bolstering the robustness and stability of the modeling processes.

3.6 Result for real case study

3.6.1 Essential key elements of the real case study

As a reminder of the real case study presented in Chapter 2, two classrooms at the Polytech Angers school were extensively equipped with multi-sensor boards. Both class-

rooms have dimensions of 7.8 meters in length and 6.6 meters in width. Specifically, Classroom 219 was equipped with three target sensors identified as (1097BD2AE2CC, 1097BD29A4A8, 1097BD2B0D44). Additionally, 14 multi-sensor electronic boards (labelled as "100" to "113") were placed in the classroom, as shown in Figure 3.15, at a height of 2.5 meters.

Each electronic board on the multi-sensor network (either target or wall sensors) can measure various parameters, including temperature, humidity, CO₂, VOCs (volatile organic compounds), brightness, and noise levels. Moreover, two sensors measure the overall power consumption of the room and the lighting power, and their data is also saved in another database through Wi-Fi communication. Additionally, window-opening sensors are installed, and a weather station has been set up on the school roof.

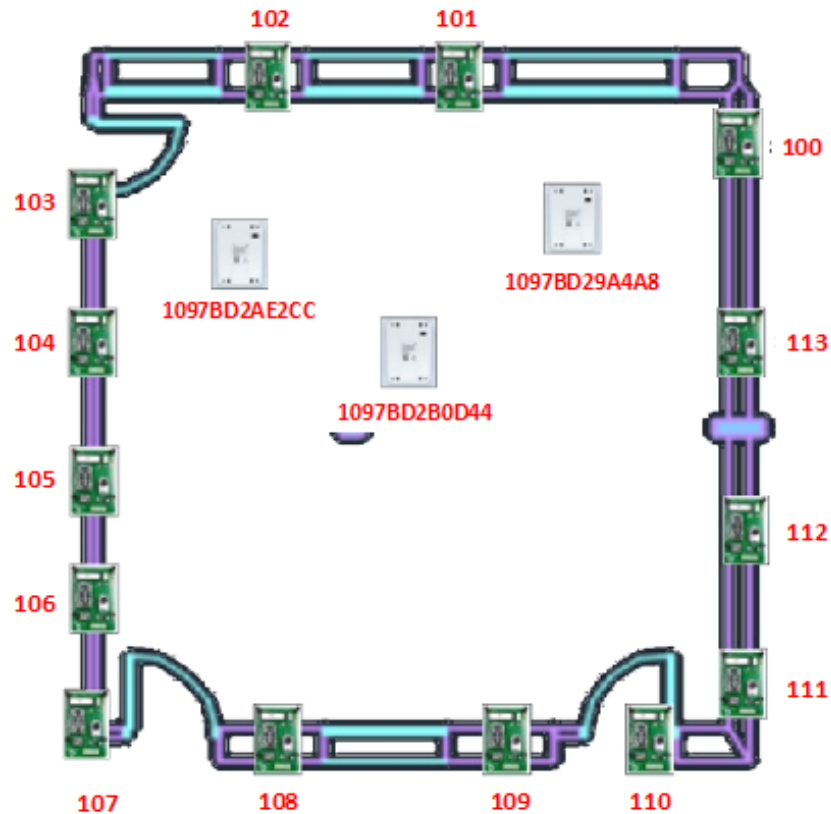


Figure 3.15 – Layout of the sensors in Real case

Due to frequent disconnections experienced, the data from the target sensors is only available for a specific time frame, ranging from October 3, 2022, at 12 noon to October 7, 2022, at 12 noon. Consequently, filtering the data from all 14 multi-sensors to match

this time frame was required. The number and percentage of missing data for each sensor can be found in Table 3.1.

Parameter	Missing Data	Percentage (%)
CO ₂ _100	2819	21.88
CO ₂ _101	2842	22.07
CO ₂ _102	2837	22.03
CO ₂ _103	2867	22.26
CO ₂ _104	2782	21.60
CO ₂ _105	2820	21.90
CO ₂ _106	2805	21.77
CO ₂ _107	2826	21.93
CO ₂ _108	11968	92.96
CO ₂ _109	11528	89.49
CO ₂ _110	11593	90.00
CO ₂ _111	2731	21.20
CO ₂ _112	2778	21.57
humidity_100	2819	21.88
humidity_101	2842	22.07
humidity_102	2837	22.03
humidity_103	2867	22.26
humidity_104	2782	21.60
humidity_105	2820	21.90
humidity_106	2805	21.77
humidity_107	2826	21.93
humidity_108	11968	92.96
humidity_109	11528	89.49
humidity_110	11593	90.00
humidity_111	2731	21.20
humidity_112	2778	21.57
temperature_100	2819	21.88
temperature_101	2842	22.07
temperature_102	2837	22.03
temperature_103	2867	22.26
temperature_104	2782	21.60
temperature_105	2820	21.90
temperature_106	2805	21.77
temperature_107	2826	21.93
temperature_108	11968	92.96
temperature_109	11528	89.49
temperature_110	11593	90.00
temperature_111	2731	21.20
temperature_112	2778	21.57

Table 3.1 – Missing data for each parameter and sensor

The target sensor data is aligned with the multi-sensor data based on the time. Sensors 108, 109, and 110 have no available data (percentage of missing data higher or equal to 90%) and are thus excluded from the analysis. For the remaining sensors, any missing

data points are imputed by taking the mean of the surrounding values. Only two target sensors, 1097B2AE2CC and 1097B29A4A8, have been finally used in this study, providing sufficient amount of data for analysis.

In summary, using the same notations as in the previous sections, our goal is to determine the $M = 3$ optimal sensor locations out of $m = 10$ candidate locations (3 sensors, 108, 109, 110, having been excluded) that enable robust and accurate prediction of temperature, humidity, and/or CO_2 levels (the sensors are multi-physical) at $n = 2$ target locations in the room. These data are collected over $T = 409$ time intervals. Data collected in site by the multi-sensors (wall and target points) for each parameters Temperature, CO_2 and Humidity are provided in Figures 3.16 to 3.18

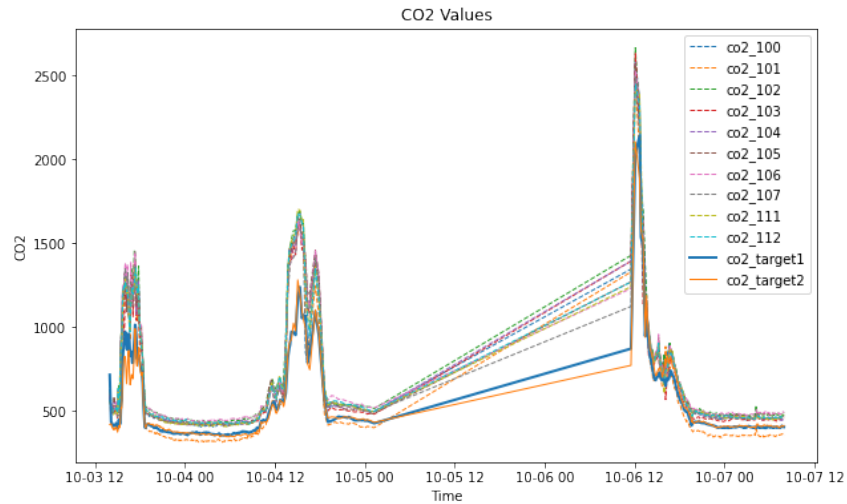


Figure 3.16 – CO_2 variation measured at different sensors and targets

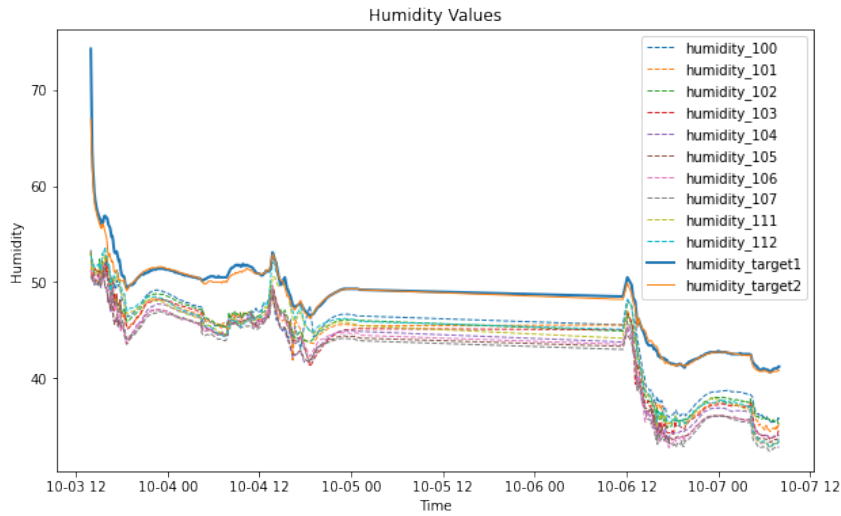


Figure 3.17 – Humidity variation measured at different sensors and targets

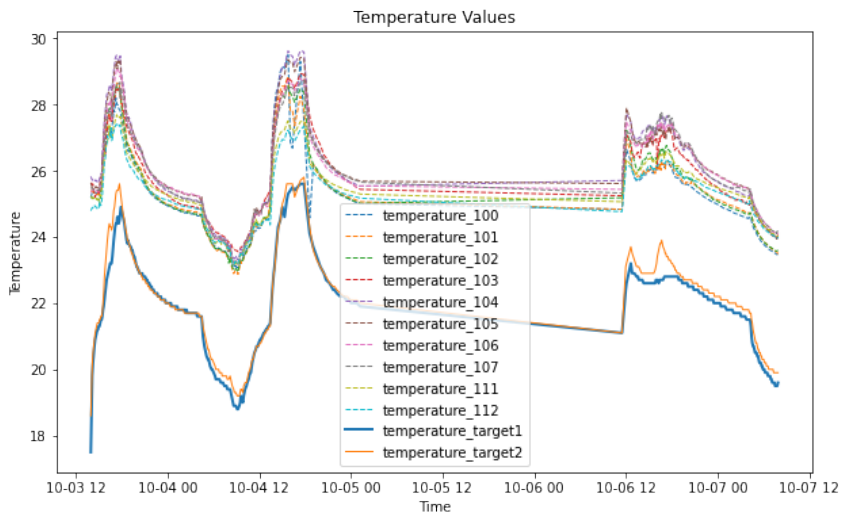


Figure 3.18 – Temperature variation measured at different sensors and targets

3.6.2 Finding the optimal sensor location

Since we have access to multi-physical data, it is essential to leverage them as much as possible to extract the maximum information and knowledge. Therefore, the EIM and IE methods used earlier will be revisited here, combining the information collected simultaneously on temperatures (T), CO₂ levels (CO₂), and humidity (H). Thus, separate matrices are created for each of these parameters – (T), (CO₂) and (H) -, containing

target point data ($[q]$) and potential sensor location data ($[U]$). Additionally, weights are assigned to each parameter, reflecting their importance in the decision-making process. For this study, the weights assigned to temperature, CO₂ concentration, and humidity are $w = [0.4, 0.4, 0.2]$, as we place higher importance on temperature and CO₂ concentration due to their potential direct connection with occupancy, respectively. These weights can be adjusted based on the relative importance of each parameter.

For the EIM, the A_0 matrix (see Equation 3.12), up to a multiplicative constant, corresponding to the Fisher information matrix, is expressed as follows:

$$A_{0_i} = (U_i \text{pinv}(q))^T \cdot (U_i \text{pinv}(q)) \quad (3.37)$$

Then, we can compute the condition number of A_0 (and of the Fisher Information Matrix) for each physical parameter p (T, CO₂ or H), and adding them using the assigned weights to form a single combined condition number for each measurement point. Next, we identify the measurement point with the lowest combined condition number and mark it as eliminated. Then, we update the priority vector to indicate the priority of this point, and remove the eliminated measurement point from the U matrices of all parameters and repeat the process until all measurement points have been considered.

For the IE, the entropy for a single parameter p is calculated as follows:

$$\text{entropy}_p = 0.5 \log \frac{\det(\text{cov}_{TOT})}{\det(\text{cov}_q)} \quad (3.38)$$

where cov_{TOT} is the covariance matrix of the combined data (q and U) for parameter p , and cov_q is the covariance matrix of the target points data (a) for parameter p .

The combined entropy for a candidate location is obtained by summing the weighted entropy for each parameter:

$$\text{entropy}_{\text{combined}} = \sum_{p=1}^{\text{num_parameters}} w(p) \cdot \text{entropy}_p \quad (3.39)$$

To find the optimal sensor locations, the combined information entropy is calculated for

all candidate locations. The locations with the highest combined entropies are considered the best choices for sensor placement.

We present in Figure 3.19 the three prioritized optimal locations selected by the EIM (framed by blue lines) and IE methods (framed by red lines) among the ten pre-defined wall sensor locations. Naturally, given the smaller number of possible candidates (10 instead of 528 or double in CFD-based simulation case studies), there is a higher probability of locations being selected by both methods. The EIM method prioritizes the selection of sensor 101, followed by 105 and 100. Information entropy methods prioritize sensor 100, followed by 101, and 103. Both algorithms choose 100 and 101 as priorities.

Observing the figure, we can see that it aligns well with the features each method emphasizes EIM favors locations with more independent information (the selected sensors are installed in a wider area), while IE method prioritize locations with ample information (the sensors are both closer to the windows, where stronger entropy variations can be captured, as well as to the target points).

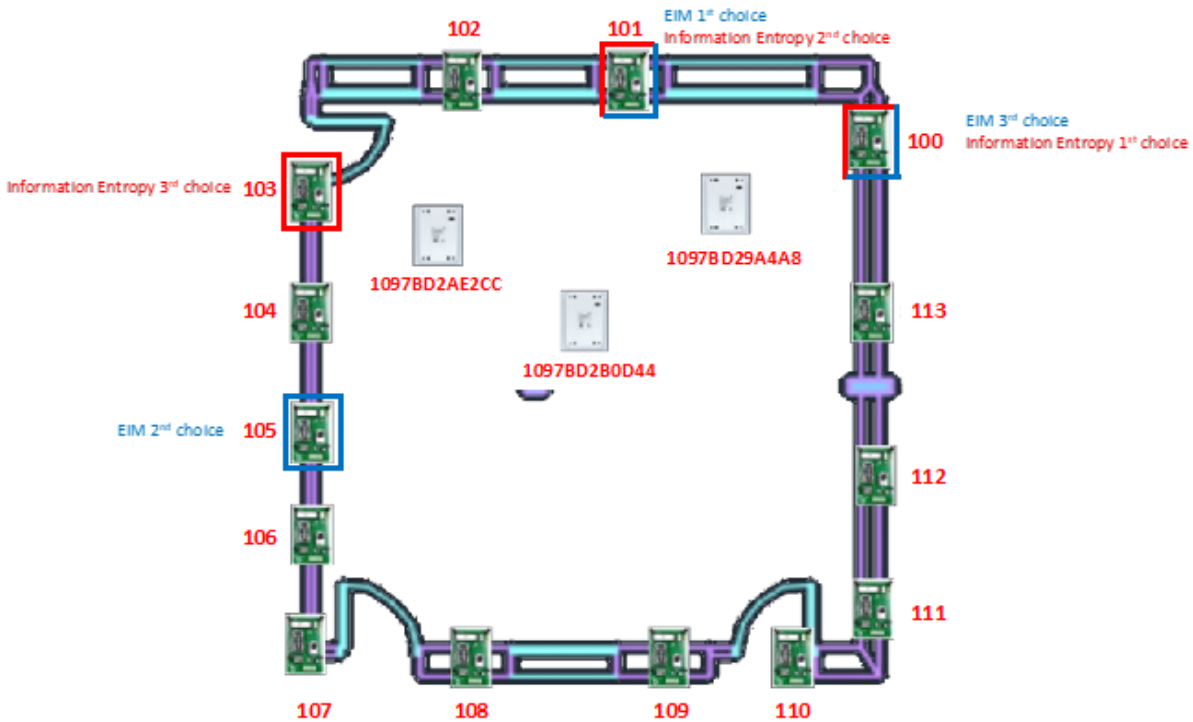


Figure 3.19 – Top three optimal sensor location for each method

In Table 3.2, the condition number, relative to the use of EIM, is calculated for each

Round	Cond. Number Temp.	Cond. Number CO ₂	Cond. Number Humidity	Total Cond. Number	Priority	Deleted Location
1	50.015	496.622	7.185	220.091	0.1	102
2	4.599	438.044	7.185	178.494	0.2	111
3	8.650	383.483	9.448	158.743	0.3	104
4	15.240	333.839	8.615	141.354	0.4	112
5	24.461	281.738	7.508	123.981	0.5	103
6	39.868	233.760	7.508	110.953	0.6	106
7	65.872	193.598	8.309	105.450	0.7	107
8	110.934	156.097	7.809	108.374	0.8	100
9	247.015	985.289	8.122	494.546	0.9	105
10	2581.029	460.338	8.122	1218.171	1.0	101

Table 3.2 – Condition numbers and priority for EIM

parameters. Reminding that a smaller condition number for a sensor means that removing that sensor causes a smaller increase in system unobservability, indicating that the sensor provides less unique information to the system. The combined condition number is calculated based on the weight of each parameters. This is used to find the sensor location that provides the least amount of unique information across all parameters (temperature, CO₂, and humidity). In other words, the sensor with the smallest combined condition number in each round is the least valuable sensor and is therefore removed first. The priority of each sensor location is determined by the round in which it is removed. The later a sensor is removed, the higher its priority, as it means that the sensor provides more unique information to the system. Therefore, the sensors with the highest priorities are the optimal locations as they contribute most to the system observability. In the table, the line in bold shows the OSP for EIM in this case.

So in summary:

- The condition number is used to measure the information contribution of each sensor.
- The combined condition number is used to determine which sensor to remove in each round.
- The priority is used to determine the optimal sensor locations. The higher the priority, the more optimal the sensor location.

In Table 3.3 for the specific use of IE method, the entropy is calculated for each parameters, and the total entropy is calculated based on the weight of each parameters,

Location	Entropy (Temperature)	Entropy (CO ₂)	Entropy (Humidity)	Total Entropy
100	-0.821	4.381	0.272	1.478
101	-0.994	4.491	0.105	1.420
102	-1.071	4.418	0.126	1.364
103	-1.116	4.449	0.257	1.385
104	-1.016	4.374	0.100	1.363
105	-1.226	4.323	0.031	1.245
106	-1.300	4.311	0.202	1.245
107	-1.152	4.255	0.270	1.295
111	-1.539	4.390	0.490	1.238
112	-1.720	4.394	0.523	1.174

Table 3.3 – Parameter entropy and total entropy

then a higher entropy value suggests a more "optimal" sensor location in the context of this code. This is because a higher entropy value indicates more uncertainty or variability in the data, which can provide more informative measurements about the state of the system. The line in bold shows the OSP for IE in this case.

3.6.3 Prediction of physical parameters at target locations using values at optimal sensor locations

To predict the physical parameters of temperature and CO₂ levels (which are presumed to be more directly related to occupancy) at the target points, we decided to explore an alternative method to those embedded in the formalisms of the EIM and IE methods.

In this section, we propose to use ANN to build individual metamodels that predict the temperature and CO₂ levels at the target points based on data available from a specific multi-sensor network. Our aim is not only to construct regression models but also to validate, through a different approach, the appropriateness of the sensors selected for the optimal locations. To achieve this, we will compare the parameter values provided by ANNs to the actual values at the target points using the RMSE as the comparison metric. The sensors that yield the lowest RMSE values will be considered the best predictors of the parameter values at the target points.

The datasets used for learning consist of sensor measurements for CO₂ levels, temperature, and humidity at ten different locations taken separately (100, 101, 102, 103, 104, 105, 106, 107, 111, and 112); thus, we have ten separate 3-by-409 (the number of time intervals) input datasets. The goal is to use these measurements to build individual ANNs

(one for each wall sensor) that separately predict the target values of CO₂ concentration and temperature¹. The output data consists of two 1-by-409 vectors (one vector for each metamodel or parameter), which are used for learning purposes (training and testing) and are identical for all sensors (i.e., for all ten separate 3-by-409 input datasets). The datasets are conventionally split into 80 percent training and 20 percent testing subsets. We emphasize that each sensor’s data is used separately for training and testing, ensuring that the trained model is specific to that sensor location.

A simple feedforward ANN with one hidden layer is employed to predict the target values. The activation functions used in the ANN play a crucial role in determining its performance. Here, the Rectified Linear Unit (ReLU) function is used as the activation function for the hidden layer neurons, while the output layer uses a linear activation function. The ANN is trained on the sensor data using backpropagation, an optimization algorithm that minimizes the error between the predicted and actual target values by adjusting the weights and biases of the network.

Figures 3.20 and 3.21 along with Tables 3.4 and 3.5 present the combined RMSE between the predicted and actual results for target sensors 1097BD29A4A8 and 1097BD2AE2CC, respectively.

For sensor 1097BD29A4A8, a close examination of Figure 3.20 reveals that sensors 100, 101, 104, 105, and 111 exhibit notably low combined RMSE values for temperature. Particularly, sensors 101, 100, and 105 display minimal combined RMSE for CO₂. Table 3.4 further quantifies these observations: Sensor 105 registers the lowest RMSE for temperature at 0.66, closely followed by sensors 101 and 103, both at 0.72, and then by sensor 100 at 0.98. For CO₂ predictions, sensor 101 stands out with an RMSE of 43.67, with sensor 105 trailing at 58.1. Although sensors 107, 111, 112, and 100 demonstrate commendable performance in predicting CO₂, the subpar performance of sensors 107 and 112 in temperature prediction cannot be overlooked. Consequently, based on a holistic evaluation, sensors 101, 105, 100, and 111 emerge as the most reliable.

1. one metamodel for each of these two parameters, humidity being excluded from the parameters of interest. Given the significance of CO₂ and temperature parameters in future occupant detection applications within the machine learning domain, these two parameters are our main focus. Although humidity is included in the sensor data, its weight is relatively lower compared to CO₂ and temperature when determining optimal sensor locations, reflecting its lesser importance in our analysis

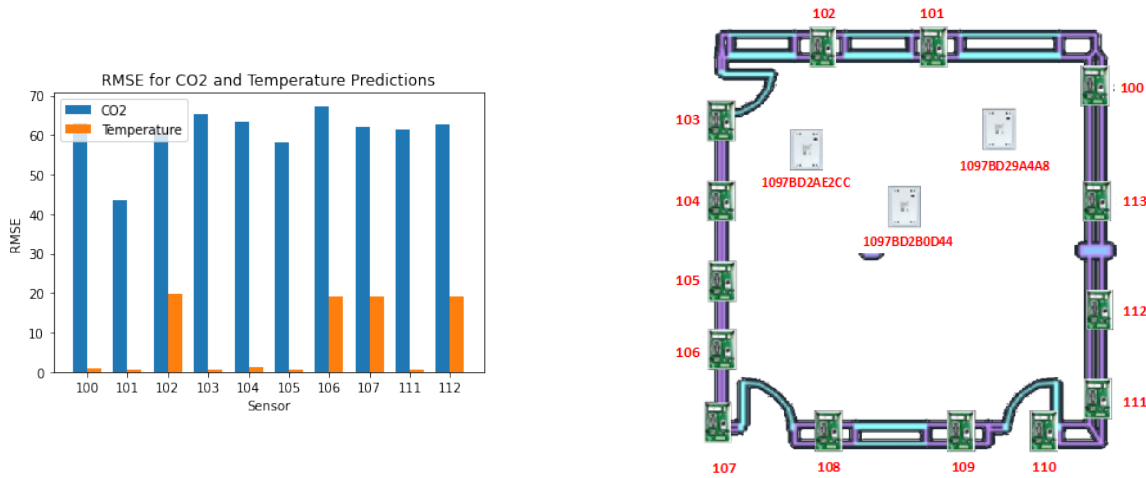


Figure 3.20 – RMSE for all sensors with regard to target point 1097BD2A4A8

Sensor	RMSE for CO2	RMSE for Temperature
100	63.09	0.98
101	43.67	0.72
102	60.44	19.68
103	65.28	0.72
104	63.49	1.13
105	58.10	0.66
106	67.35	19.25
107	62.19	19.07
111	61.50	0.73
112	62.59	19.15

Table 3.4 – Combined RMSE for each wall sensor to target point 1097BD2A4A8

For sensor 1097BD2AE2CC, Figure 3.21 indicates superior performance by sensors 100, 101, 105, and 111. Focusing on the results presented in Table 3.5, sensor 105 has the most accurate temperature predictions with an RMSE of 0.5. This is followed closely by sensors 101 and 104, both at 0.58, then sensor 111 at 0.61, sensor 103 at 0.75, and sensor 100 at 1.02. In terms of CO₂ predictions, sensor 101 leads with an RMSE of 57.86, succeeded by sensor 106 at 60.84. Sensors 105, 107, 111, 112, and 100 all cluster around an RMSE value of 80. Factoring in performance across both metrics, sensors 101, 105, 111, and 100 distinguish themselves as the foremost choices.

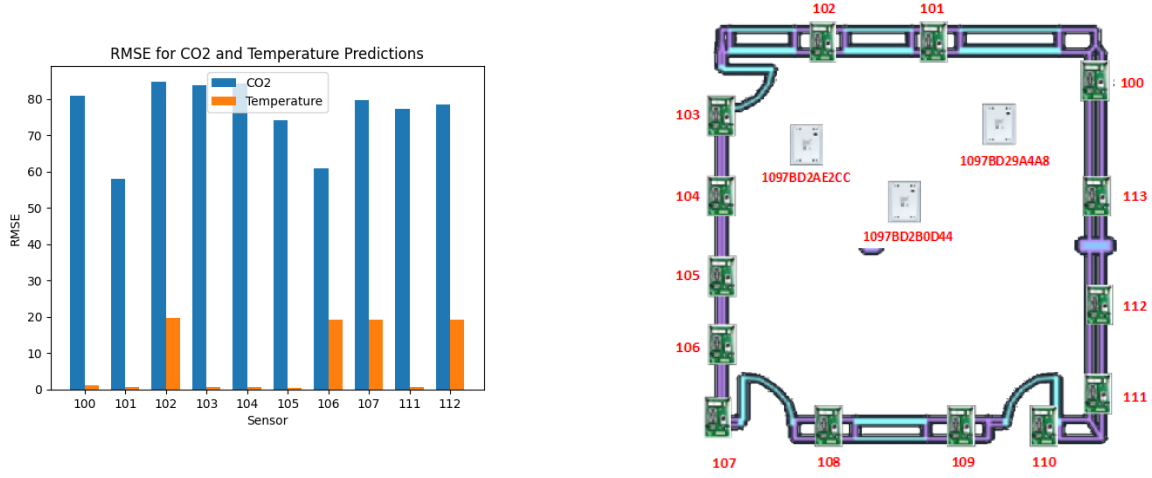


Figure 3.21 – RMSE for all sensors with regard to target point 1097BD2E2CC

Sensor	RMSE for CO2	RMSE for Temperature
100	80.86	1.02
101	57.86	0.58
102	84.77	19.71
103	83.85	0.75
104	84.26	0.58
105	74.00	0.50
106	60.84	19.27
107	79.71	19.08
111	77.35	0.61
112	78.35	19.16

Table 3.5 – Combined RMSE for each wall sensor to target point 1097BD2E2CC

For a general assessment, Figures 3.22 to 3.41 show a comparison between actual and predicted values of parameters (CO₂ level and Temperature). From the figures, we can notice the good performance for the 101, 105, 100 and 111 as well. The cumulative results unequivocally highlight that sensor 101 consistently exhibits the lowest RMSE for both CO₂ and Temperature, signifying its superior performance. This is closely followed by sensors 105, 100, and 111 in that order. Notably, the EIM method prioritizes sensor 101 as its primary choice, with sensors 105 and 100 as its subsequent selections. On the other hand, the IE method ranks sensor 100 as its top pick, followed by sensor 101. Given this

alignment in sensor rankings between empirical RMSE results and the two methodologies, we can assert with confidence that our approach is both robust and compelling.

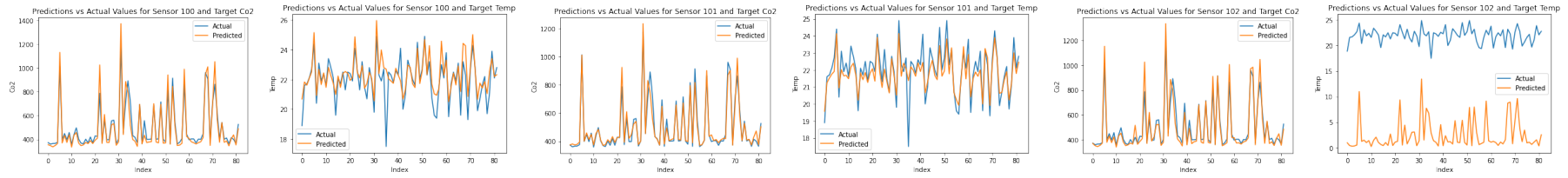


Figure 3.22 – Sensor 100 CO₂ VS target CO₂
 Figure 3.23 – Sensor 100 Temperature VS target Temperature
 Figure 3.24 – Sensor 101 CO₂ VS target CO₂
 Figure 3.25 – Sensor 101 Temperature VS target Temperature
 Figure 3.26 – Sensor 102 CO₂ VS target CO₂
 Figure 3.27 – Sensor 102 Temperature VS target Temperature

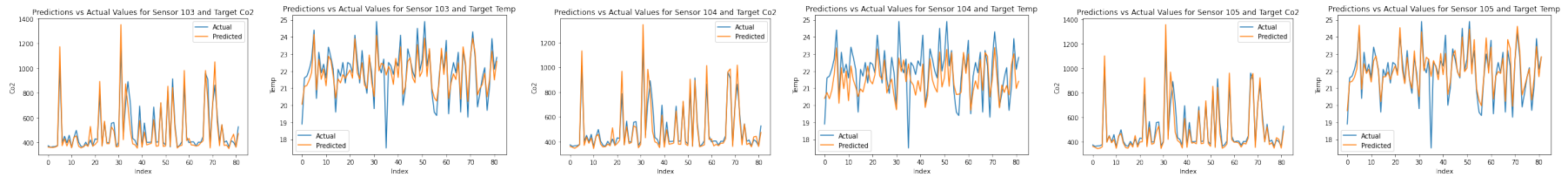


Figure 3.28 – Sensor 103 CO₂ VS target CO₂
 Figure 3.29 – Sensor 103 Temperature VS target Temperature
 Figure 3.30 – Sensor 104 CO₂ VS target CO₂
 Figure 3.31 – Sensor 104 Temperature VS target Temperature
 Figure 3.32 – Sensor 105 CO₂ VS target CO₂
 Figure 3.33 – Sensor 105 Temperature VS target Temperature

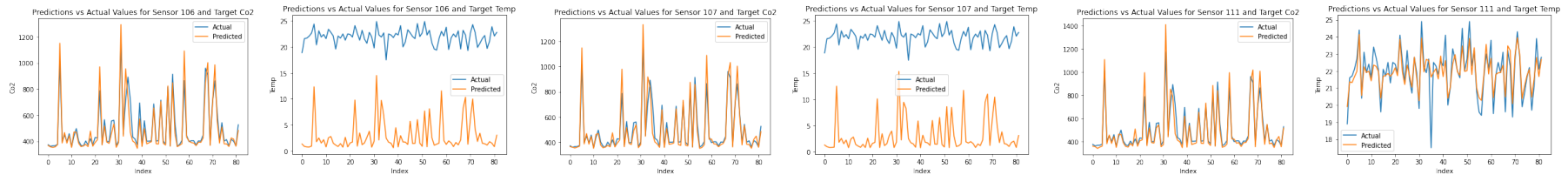


Figure 3.34 – Sensor 106 CO₂ VS target CO₂
 Figure 3.35 – Sensor 106 Temperature VS target Temperature
 Figure 3.36 – Sensor 107 CO₂ VS target CO₂
 Figure 3.37 – Sensor 107 Temperature VS target Temperature
 Figure 3.38 – Sensor 111 CO₂ VS target CO₂
 Figure 3.39 – Sensor 111 Temperature VS target Temperature



Figure 3.40 – Sensor 112 CO₂ VS target CO₂

Figure 3.41 – Sensor 112 Temperature VS target Temperature

3.7 Conclusion of the chapter

In this chapter, we focused on the complexity of the optimal sensor location problem, which holds significant importance in various scientific, engineering, and industrial applications. Our thorough examination aimed to provide a clear overview of the problem, its various aspects, and the state-of-the-art solutions developed to address it.

After carefully evaluating different methods, we have chosen the EIM and IE methods and have successfully transferred them for the first time to the building indoor environment monitoring sensor placement problem. We provide a comprehensive introduction to both methods, explaining their basic principles, their operation, and their advantageous conditions.

To validate our choices and understand their real-world performance, we applied both methods to simulated data, closely mimicking actual deployment scenarios. The optimality of the selected sensor locations was then validated using two distinct parameters: model stability, examining how robust the models are against input variations, and RMSE, quantifying the accuracy of sensor readings compared to actual target values.

Our research also expanded into complex scenarios, exploring multi-zone and multi-wall environments. This investigation was designed to test the adaptability and versatility of our chosen methods, as real-world applications often involve complex and variable environments.

Furthermore, we moved beyond simulations and applied the two methods to real-world data, validating the results using prediction and RMSE. These real-world tests provided concrete evidence of how well our methods performed under actual operating conditions, demonstrating their reliability.

However, it is essential to consider that 'optimal' is a term defined within the context of a specific application. While our results are promising, they should be interpreted within the parameters of our study. Future work should further explore and refine these methods, extending their applicability to a broader range of environments and conditions.

SINGLE EVENT DETECTION

As previously discussed in earlier chapters of this thesis, the field of activity detection in buildings is rapidly evolving, focusing on the automatic identification and classification of human behavior within indoor environments. In this chapter, our attention turns to machine learning techniques, where we will undertake a comparative analysis of the advantages and disadvantages associated with various approaches, including supervised methods such as Logistic Regression, ANN, and XGBoost, as well as semi-supervised techniques like AutoEncoder, and unsupervised methods such as DBSCAN, PCA, and Mixture of Gaussian Distributions (MGD). These machine learning approaches will be applied to both simulated (for highly imbalanced and more realistic occupancy scenario) and real-time data for the detection of window status and the presence of occupants. After conducting a thorough analysis and testing, it becomes evident that DBSCAN and XGBoost emerge as the most suitable algorithms for our purposes. In the final stages, we also explore techniques for accelerating these algorithms. Notably, we achieve a significant enhancement in computation time for DBSCAN, which is now ten times faster than its previous iteration. Additionally, parameter grid search computation time is halved. In the case of XGBoost, computation time is improved, resulting in an 85% reduction.

4.1 Introduction

In Chapter 1, we proposed a state-of-the-art literature, showing remaining challenges in using machine learning techniques for occupancy detection. The primary challenge is the presence of imbalanced datasets. Anomalies or rare activities, essential for detection, often represent a small fraction of the data. This imbalance can skew algorithms towards the majority class, compromising their ability to identify anomalies effectively. Strategies such as resampling techniques have been explored to mitigate this issue and improve detection accuracy, as discussed in previous research.

Acquiring labeled data, especially for abnormal behavior, remains the second main

challenge. Supervised machine learning techniques necessitate labeled data for training, which is often scarce and expensive to obtain. In response, innovative approaches like NormA (Boniol et al.2021) and others have emerged, offering unsupervised and scalable subsequence anomaly detection solutions. These methods alleviate the need for extensive domain-specific knowledge and open doors to broader applications.

The third challenge is the adaptability to dynamic environments. Real-world data is subject to changes in patterns and conditions over time. Static anomaly detection models struggle to adapt, resulting in reduced performance. Recognizing the importance of adaptability, recent research has proposed dynamic ensemble algorithms and incremental learning techniques, which demonstrate promise in handling evolving data streams with various types of changes.

The fourth challenge is the high false positive and false negative rates. Achieving a balanced trade-off between these rates is critical, particularly in applications with high stakes, such as healthcare and industrial maintenance. Research has delved into the development of reliable and accurate anomaly detection techniques, emphasizing their significance in real-world scenarios.

Building on these challenges, the present chapter seeks to refine anomaly detection in building IoT systems. We will address these challenges systematically, leveraging optimal sensor selection and data fusion techniques to improve accuracy. Strategies for handling imbalanced datasets, privacy-preserving methods relying solely on environmental sensor data, and the exploration of unsupervised and semi-supervised algorithms will be at the forefront of our investigation.

Drawing a connection between the machine learning methods discussed in Chapter 1 and the challenges we aim to address here in the context of occupant activity detection, we have emphasized the importance of studying the seven algorithms summarized in Table 4.1 below.

Method Type	Algorithms
Unsupervised	DBSCAN, PCA, MGD
Supervised	Logistic Regression, ANN, XGBoost
Semi-supervised	AutoEncoder

Table 4.1 – Machine Learning Methods

By combining these seven algorithms appropriately and tuning their parameters, it is possible to develop a robust anomaly detection system capable of effectively addressing

the challenges of activity detection in buildings.

4.2 Methodologies detailed

4.2.1 Unsupervised DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is an unsupervised clustering algorithm, proposed by Ester et al. (1996), which can serve as a tool for unsupervised anomaly detection. Unlike partitioning and hierarchical clustering methods, DBSCAN is designed to discover clusters of arbitrary shape in datasets containing noise and outliers.

Considering the four challenges highlighted in the introduction of the present chapter, despite being an unsupervised algorithm, DBSCAN can be adapted to imbalanced data by appropriately setting parameters to identify anomalies as isolated or low-density points. This is helpful in detecting rare events in the presence of mostly normal data. It can also adapt to dynamic environments by identifying clusters of points based on density. When new data is introduced, DBSCAN can adjust clusters based on density changes, making it suitable for evolving environments. And generally, DBSCAN, is known to contribute to reducing false positives by identifying anomalies based on density.

In practice, DBSCAN operates on the principle that clusters in a dataset correspond to dense regions separated by areas of lower density. The algorithm classifies data points into three categories: core points, border points, and noise points (anomalies). These classifications are essential for understanding how DBSCAN identifies window opening cases within our context.

The key terms used in DBSCAN include:

1. **Neighborhood:** The neighborhood of a point p in DBSCAN is simply the set of points that lie within a distance eps from p . DBSCAN uses this concept to understand the density of points around a given point. If there are $minPts$ within the eps -neighborhood of a point, that point is considered a "core" point. The eps -neighborhood of a point p in the database D is defined as:

$$N_{eps}(p) = \{q \in D | dist(p, q) \leq eps\} \quad (4.1)$$

where $dist(p, q)$ is a function returning the distance between points p and q .

2. **eps**: *eps* is a user-defined parameter¹ that specifies the maximum distance between two points for them to be considered as in the same neighborhood. The optimal value of *eps* varies based on the dataset and should be chosen such that it best meets the objective of the analysis. One common approach is to use a k-distance graph to find the best *eps*.
3. **minPts**: *minPts* is a user-defined parameter¹ that specifies the minimum number of points required to form a dense region. Like *eps*, the optimal value of *minPts* can depend on the dataset, and it is often chosen based on domain knowledge. A common rule of thumb is to set *minPts* to twice the dimensionality of the dataset, although this may not be optimal in all cases.
4. **Directly Density-Reachable**: In the DBSCAN algorithm, a point p is directly density-reachable from a point q if p is within the *eps*-neighborhood of q , and q is a core point (i.e., q has at least *minPts* within its *eps*-neighborhood). This concept is used during the expansion of clusters. When a core point is found, all points that are directly density-reachable from the core point (i.e., all points within its *eps*-neighborhood) are added to the same cluster.
5. **Density-Reachable**: A point p is density-reachable from a point q if there is a sequence of points p_1, \dots, p_n such that $p_1 = q$, $p_n = p$, and each point p_{i+1} is directly density-reachable from p_i . This concept is used to add points to existing clusters. If a point is density-reachable from any point in a cluster, it is added to that cluster.
6. **Density-Connected**: A point p is density-connected to a point q if there is a point o such that both p and q are density-reachable from o . This concept ensures that all points in a cluster are connected. That is, for any two points in a cluster, there is a chain of points within the cluster such that each point in the chain is directly density-reachable from the previous point. This results in a set of points that are all density-connected, forming a cluster.
7. **Cluster**: A cluster C with respect to *eps* and *minPts* is a non-empty subset of the database D satisfying the following conditions:
 - For all points p, q : if p is in C and q is density-reachable from p with respect to *eps* and *minPts*, then q is in C (Maximality).

1. domain expertise can help in setting these parameters initially. If we know the scale at which we expect clusters to appear or how dense they should be, this can guide the choice.

- For all points p, q in C : p is density-connected to q with respect to eps and $minPts$ (Connectivity).

These concepts are utilized in the DBSCAN's cluster formation process and the identification of core, border, and noise points. Core points are central to cluster formation, while border points surround core points, and noise points do not belong to any cluster.

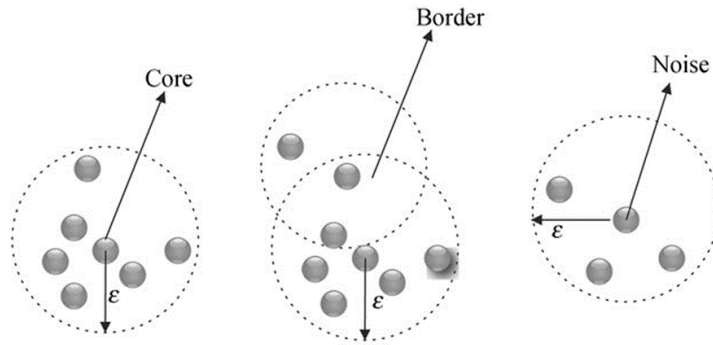


Figure 4.1 – Illustration of core, border and noise points associated with DBSCAN algorithm (from Amini et al. (2014)).

To adapt DBSCAN to our window opening detection (considered as anomalies) context, we follow these steps:

1. **Initialize:** Starting with an arbitrary unvisited point in the dataset and retrieve its eps -neighborhood, which includes all points within a distance of eps from the selected point.
2. **Check for Core Point:** Determine if the selected point qualifies as a core point. A point is a core point if there are at least $minPts$ in its eps -neighborhood, including the point itself. In the context of our case study, a core point can be a data point where the window is closed and shares similar feature values with its neighboring points. If it meets the criteria of a core point, a new cluster is created, and all points in the eps -neighborhood are added to this cluster.
3. **Expand the Cluster:** If the initial point is a core point, iteratively check all the points within the eps -neighborhood of the initial point and include nearby core points to the cluster.
4. **Identify Border Points:** While the algorithm progresses through the points in the eps -neighborhood of the initial core point, it may encounter points that are

not core points themselves but fall within the *eps*-neighborhood of a core point. These points are classified as border points and are added to the cluster associated with the nearby core point. In our case study, a border point might represent a data point where the window is closed but exhibits slightly different feature values compared to a nearby core point, possibly due to minor variations in environmental conditions.

5. **Repeat:** Repeat the process for all unvisited points in the dataset. If a point fails to meet the criteria of a core point and does not fall within the *eps*-neighborhood of any core point, it is categorized as noise.
6. **Anomaly Detection:** Once all points have been visited, and clusters have been formed, points labeled as noise are considered anomalies. In our case study, the anomalies would likely instances where the window is open. These events are rare (only 147 occurrences out of 35,040 time steps of our in the very imbalanced simulated case study) and exhibit distinctive feature values compared to the remainder of the dataset. These points does not belong to any cluster and do not share proximity with the denser regions of the dataset to qualify as border points.

The pseudocode of DBSCAN is shown as follows.

Algorithm 3 DBSCAN-Anomaly-Detection

```

1: function DBSCAN-ANOMALY-DETECTION( $D, eps, minPts$ )
2:    $C = 0$ 
3:    $Noise = \{\}$ 
4:   for each unvisited point  $P$  in dataset  $D$  do
5:     mark  $P$  as visited
6:      $NeighborPts = regionQuery(P, eps)$ 
7:     if  $sizeof(NeighborPts) < minPts$  then
8:       add  $P$  to  $Noise$ 
9:     else
10:       $C = nextCluster()$ 
11:       $expandCluster(P, NeighborPts, C, eps, minPts)$ 
12:    end if
13:  end for
14:  return  $Noise$ 
15: end function
16: function EXPANDCLUSTER( $P, NeighborPts, C, eps, minPts$ )
17:  add  $P$  to cluster  $C$ 
18:  for each point  $P'$  in  $NeighborPts$  do
19:    if  $P'$  is not visited then
20:      mark  $P'$  as visited
21:       $NeighborPts' = regionQuery(P', eps)$ 
22:      if  $sizeof(NeighborPts') \geq minPts$  then
23:         $NeighborPts = NeighborPts$  joined with  $NeighborPts'$ 
24:      end if
25:    end if
26:    if  $P'$  is not yet a member of any cluster then
27:      add  $P'$  to cluster  $C$ 
28:    end if
29:  end for
30: end function
31: function REGIONQUERY( $P, eps$ )
32:  return all points within  $P$ 's  $eps$ -neighborhood (including  $P$ )
33: end function

```

4.2.2 Unsupervised Principal Component Analysis (PCA)

PCA, first proposed by Karl Pearson in 1901, is a widely used unsupervised learning algorithm in machine learning and statistics. PCA's primary objective is to reduce the dimensionality of a high-dimensional dataset while retaining as much of the data's variation as possible. It accomplishes this by identifying new uncorrelated variables known as principal components, which are linear combinations of the original variables. These principal components are ordered so that the first few capture most of the variation present in all of the original variables.

PCA offers solutions to several challenges discussed in the introduction. It can assist in handling imbalanced data by reducing data dimensionality while preserving important information, thereby creating a more balanced dataset in a projection space where anomalies may become more apparent. Additionally, PCA can adapt to dynamic environments by recalculating principal components as new data arrives, maintaining an efficient representation of evolving data. Similar to DBSCAN, PCA can contribute to reducing false positives by identifying anomalies based on unusual data characteristics, thus increasing anomaly detection specificity.

When given a dataset represented as a $d \times n$ matrix \mathbf{X} , where d is the number of dimensions (features) and n is the number of observations, PCA aims to find a set of d orthogonal vectors (principal components) that can optimally reconstruct the original data. These vectors are the eigenvectors of the covariance matrix of \mathbf{X} , corresponding to its largest eigenvalues.

The steps to perform PCA are as follows:

1. **Standardize the dataset:** PCA is sensitive to the scales of the variables. If the scales are not similar, standardizing the features to have zero mean and unit variance is often crucial preprocessing step.
2. **Compute the covariance matrix:** Calculate the covariance matrix, denoted as \mathbf{C} , is a $d \times d$ matrix where each element represents the covariance between two features. The covariance between two features x_i and x_j is calculated as:

$$C_{ij} = \frac{1}{n-1} \sum_{k=1}^n (x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j) \quad (4.2)$$

where \bar{x}_i and \bar{x}_j are the means of the features x_i and x_j respectively.

3. **Compute the eigenvectors and eigenvalues of the covariance matrix:** The eigenvectors (principal components) of the covariance matrix are orthogonal vectors that define the new feature space. The eigenvalues represent the variance of the data along the corresponding eigenvectors. In other words, they give the "importance" of different features in the dataset.
4. **Sort the eigenvectors:** Arrange the eigenvectors in decreasing order of their corresponding eigenvalues. The sorted eigenvectors form a matrix \mathbf{P} that is used for projecting the original data onto the new feature space.
5. **Project the original data:** Transform the original dataset \mathbf{X} into the new feature space \mathbf{Y} by multiplying it with the projection matrix \mathbf{P} , resulting in a dataset of reduced dimensions.

$$\mathbf{Y} = \mathbf{P}^T \mathbf{X} \quad (4.3)$$

PCA is effectively used for unsupervised anomaly detection. The core idea is that normal data instances (non-anomalies) will be closer to the subspace spanned by the principal components, while anomalies are more likely to be distant from this subspace. Hence, the distance or error of an instance when projected onto the subspace serves as an anomaly score.

One approach to compute anomalies is to reconstruct the original dataset from the projected data (after first obtaining the principal components and projecting data). This can be achieved by multiplying the projected data with the transpose of the projection matrix:

$$\mathbf{X}_{rec} = \mathbf{P} \mathbf{Y} \quad (4.4)$$

\mathbf{X}_{rec} may not be identical to the original dataset \mathbf{X} , especially if some dimensions (principal components) were discarded during dimensionality reduction. The difference between the original data and the reconstructed data, often measured by the mean squared error (MSE), serves as the anomaly score:

$$MSE = \frac{1}{n} \sum_{i=1}^n (\mathbf{X}_i - \mathbf{X}_{rec,i})^2 \quad (4.5)$$

where \mathbf{X}_i and $\mathbf{X}_{rec,i}$ are the i -th instances in the original and reconstructed datasets,

respectively. Anomalies are instances with high anomaly scores, indicating poor reconstruction by the PCA model. A common way to identify anomalies is by setting a threshold on the anomaly score, which can be defined in various ways, such as a fixed value, a specific percentile of the anomaly scores (e.g., 99th percentile), or a certain number of standard deviations from the mean anomaly score.

4.2.3 Unsupervised Multivariate Gaussian distribution (MGD)

The MGD proves to be a robust tool for anomaly detection, particularly in scenarios involving multidimensional data. It excels at modeling the probability distribution of datasets in multidimensional spaces, effectively capturing correlations between various features.

In the light of the four challenges outlined in the introduction, MGD can be tailored to address imbalanced data issues by adjusting the mixture of Gaussian distribution. This adaptation allows MGD to better represent data patterns where anomalies are rare compared to normal data. Furthermore, MGD is adaptable to changing environments, as it can dynamically adjust its Gaussian distribution parameters with the introduction of new data. This feature enables it to model dynamic environments effectively and track changes in data structure over time.

For d -dimensional random vector $X = [X_1, X_2, \dots, X_d]^T$, the MGD of X is given by the following probability density function (pdf):

$$f(x; \mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right) \quad (4.6)$$

where, - $x = [x_1, x_2, \dots, x_d]^T$ is a real d -dimensional column vector. - $\mu = [\mu_1, \mu_2, \dots, \mu_d]^T$ is a d -dimensional mean vector. - Σ is a $d \times d$ covariance matrix, and $|\Sigma|$ is the determinant of Σ .

The parameters of the MGD are estimated using Maximum Likelihood Estimation (MLE). The MLE for mean vector μ and covariance matrix Σ are given by:

$$\mu = \frac{1}{n} \sum_{i=1}^n x^{(i)} \quad (4.7)$$

$$\Sigma = \frac{1}{n} \sum_{i=1}^n (x^{(i)} - \mu)(x^{(i)} - \mu)^T \quad (4.8)$$

where $x^{(i)}$ represents the i^{th} training example and n is the number of training examples. The covariance matrix Σ is a crucial element of this distribution, as it not only provides variances for individual variables (along the diagonal) but also captures correlations between variables (off-diagonal elements).

With our objective to use MGD for detection purpose, we follow these steps:

1. **Model Training:** First, the parameters of the MGD, namely the mean vector μ and the covariance matrix Σ , are estimated from the training data using the Maximum Likelihood Estimation (MLE) method.
2. **Anomaly Score Calculation:** For a new instance in the test set, its anomaly score is computed using the probability (pdf) of the trained MGD. Instances with the lower probabilities are more likely to be anomalies. Specifically, for an instance x , its anomaly score is computed using the probability density function of the MGD:

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right) \quad (4.9)$$

3. **Anomaly Determination:** If the anomaly score of an instance falls below a pre-defined threshold, it is labeled as an anomaly.

One notable advantage of this method lies in its ability to capture correlations among different features, which proves valuable when individual features are not independently distributed. However, it is essential to note that obtaining an accurate estimate of the covariance matrix can be challenging, particularly when dealing with high-dimensional data and limited data points.

4.2.4 Supervised Logistic regression

Logistic Regression is a statistical method primarily used for binary classification problems. This algorithm predicts the probability of an event's occurrence by fitting data to a logistic function, hence its name. As it deals with probabilities, its output values lie between 0 and 1. In scenarios involving imbalanced data, logistic regression can be coupled with sampling techniques such as oversampling (e.g., SMOTE, see Chapter 1, section 1.3.5) or undersampling to address the data imbalance challenge. Similar to other supervised algorithms, logistic regression can be fine-tuned to minimize false positives

and false negatives. This optimization involves adjusting decision thresholds, employing hyperparameter tuning methods, and selecting relevant features.

The fundamental formula for Logistic Regression is encapsulated in the logistic function, also known as the sigmoid function. This function maps any real-valued numbers into a range between 0 and 1, and its formula is as follows:

$$P(Y = 1|X) = \frac{1}{1 + e^{-z}} \quad (4.10)$$

Here, z is determined by $z = \beta_0 + \beta_1 X$ and $P(Y = 1|X)$ represents the probability that the class is 1 given the predictor variable X . The z equation describes a straight line, and when plotted, it resembles an “*S*”-shaped curve. In this context, ‘ X ’ serves as the input to the function, and the output falls within the 0 to 1 range. If the output surpasses 0.5, the classification outcome is typically labeled as 1 (or YES), while values below 0.5 are classified as 0 (or NO).

In the context of anomaly detection, the logistic regression model is trained using labelled dataset with the binary target variable indicating whether an instance is normal or anomalous. The logistic regression model learns to establish the boundary that separates normal instances from anomalies. Once trained, the model can predict whether a new instance is an anomaly by checking on which side of the boundary it falls.

4.2.5 Supervised Artificial neural network

ANN represent a class of machine learning models inspired by the biological neural networks. These models are renowned for their remarkable flexibility and ability to approximate a wide array of functions, rendering them exceptionally valuable across various applications, including anomaly detection. To tackle data imbalances, ANNs can effectively leverage oversampling or undersampling techniques. Similar to Logistic Regression, ANNs can also undergo optimization to discern intricate patterns and minimize classification errors.

The fundamental building block of an ANN is the artificial neuron or node. Each node accepts a set of input values, applies weighted summation to these inputs, incorporates a bias term, and subsequently applies an activation function to produce an output.

The nodes are organized into layers: an input layer, one or more hidden layers, and an output layer. The input layer receives the input data, the hidden layers perform computations, and the output layer generates the final network output. This arrangement,

where information flows sequentially from one layer to the next, gives rise to the term “feedforward neural network”, the most prevalent ANN architecture.

The ANN’s weights and biases are learned through a process called backpropagation. This process involves tracing the network’s output error backward through the layers to iteratively adjust the weights and biases, ultimately minimizing this error.

For anomaly detection, an ANN can be trained to recognize the normal data behavior. Anomalies are then identified as instances significantly deviating from this learned norm. In a supervised context, this entails training the ANN on a labeled dataset, where each instance is categorized as either normal or anomalous. Post-training, the ANN can be utilized to predict whether a new, unseen instance falls within the normal or anomalous category. This prediction is achieved by inputting the instance into the network’s input layer, allowing values to propagate forward through the layers, and extracting the network’s output as the predicted label for the instance.

4.2.6 Supervised XGBoost

XGBoost builds upon the concept of boosting, an ensemble meta-algorithm in machine learning aimed at reducing bias and variance. Boosting combines multiple "weak learners," which are models slightly better than random guessing, to create a robust predictive model. In the case of XGBoost, these weak learners take the form of decision trees. Each tree strives to correct errors or enhance predictive capabilities where previous trees may have failed. This work of optimization is founded on the objective function denoted as “ L ” which provides a quantifiable metric that the algorithm seeks to minimize during the training process.

$$L(\phi) = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \Omega(f_k) \quad (4.11)$$

The objective function $L(\phi)$ is composed of two terms:

1. $\sum_{i=1}^n l(y_i, \hat{y}_i)$: This term represents the loss function, which quantifies how well the model’s predictions \hat{y}_i match the actual data y_i . The algorithm aims to minimize this loss.
2. $\sum_{k=1}^K \Omega(f_k)$: This is the regularization term, which penalizes the complexity of the overall model.

The inclusion of a regularization term ($\Omega(f)$) in the objective function helps to control the complexity of the individual trees f_k . Regularization avoids the risk of overfitting,

where the model learns the training data too well but performs poorly on unseen data.

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \|w\|^2 \quad (4.12)$$

In this equation, T represents the number of terminal nodes (or leaves), w is the vector of scores on the leaves, and γ and λ are regularization parameters that control tree complexity and leaf scoring, respectively.

XGBoost relies on first (g_i) and second-order (h_i) derivatives of the loss function to find the best splits and leaf values:

$$g_i = \frac{\partial l(y_i, \hat{y}_i)}{\partial \hat{y}_i} \quad (4.13)$$

$$h_i = \frac{\partial^2 l(y_i, \hat{y}_i)}{\partial \hat{y}_i^2} \quad (4.14)$$

These partial derivatives are important because they are used to find the optimal split points and leaf node values that minimize the objective function $L(\phi)$.

The key steps in the XGBoost algorithm include:

1. **Initialization:** The model starts with a constant prediction value which can be set based on the proportion of instances of that class relative to all others for each class k in a problem with K classes.
2. **Gradient Computation:** In each iteration, first and second-order gradients are computed based on the current predictions and the loss function.
3. **Tree Construction:** A new tree is built using these gradients to find the optimal splits and leaf values.
4. **Model Update:** The model is updated with the newly constructed tree.
5. **Iteration:** Steps 2-4 are repeated until a predefined stopping criterion is met.
6. **Prediction:** For making predictions, the sum of scores from each tree is used.

XGBoost is a powerful method for classifying window states. It combines weak learners and an optimized objective function $L(\phi)$ with regularization terms ($\Omega(f)$) to control model complexity, which helps prevent overfitting. In simple terms, XGBoost strikes a

balance between accurate predictions and model complexity, making it effective for classification tasks. XGBoost simplifies the process of building decision trees by enabling concurrent / parallel execution of certain parts, enhancing scalability, in contrast to traditional decision trees built sequentially. A critical computational challenge in tree-based algorithms is identifying the best split points for nodes. XGBoost employs a faster approximate algorithm, outperforming exact algorithms in efficiency. The regularization parameters in XGBoost encourage simpler trees, averting overfitting and facilitating parallel tree construction for increased efficiency.

Additionally, XGBoost stores data efficiently in-memory using a structure called a "column block." This organization enables efficient data access patterns, further speeding up the algorithm. Finally, XGBoost excels at handling sparse data by efficiently bypassing missing or zero values, thereby reducing computation time.

4.2.7 Semi-supervised AutoEncoder

AutoEncoders, a type of artificial neural network, are used to efficiently represent input data. They work in an unsupervised manner but can be applied in a semi-supervised setting, particularly for tasks like anomaly detection where one type of data (e.g., normal) is common, and the other (e.g., anomalies) is scarce.

An AutoEncoder has two main components: an encoder and a decoder. The encoder transforms input data into a lower-dimensional representation or 'code', while the decoder maps this code back to the original data dimension.

For an input $x \in \mathbb{R}^n$, the encoder and decoder are represented as follows:

$$h = f_{\theta}(x) \quad (\text{encoder}) \quad (4.15)$$

$$x' = g_{\phi}(h) \quad (\text{decoder}) \quad (4.16)$$

Here, $h \in \mathbb{R}^d$ is the encoded version of x , x' is the reconstructed input, f_{θ} and g_{ϕ} are learned non-linear transformations parameterized by θ and ϕ respectively, during training.

The training process involves finding the parameters θ and ϕ that minimize a loss function, typically measuring the difference between the original input x and its reconstruction x' :

$$L(x, g_\phi(f_\theta(x))) \quad (4.17)$$

The mean squared error is commonly used as the loss function:

$$L(x, x') = \|x - x'\|^2 \quad (4.18)$$

In the context of anomaly detection, AutoEncoders are trained on 'normal' data. The idea is that they will accurately reconstruct 'normal' instances while failing to do so for anomalous ones. Anomalies are identified by measuring the reconstruction error of an instance; a high error indicates an anomaly. The entire process for AutoEncoder-based anomaly detection is illustrated in Figure 4.2

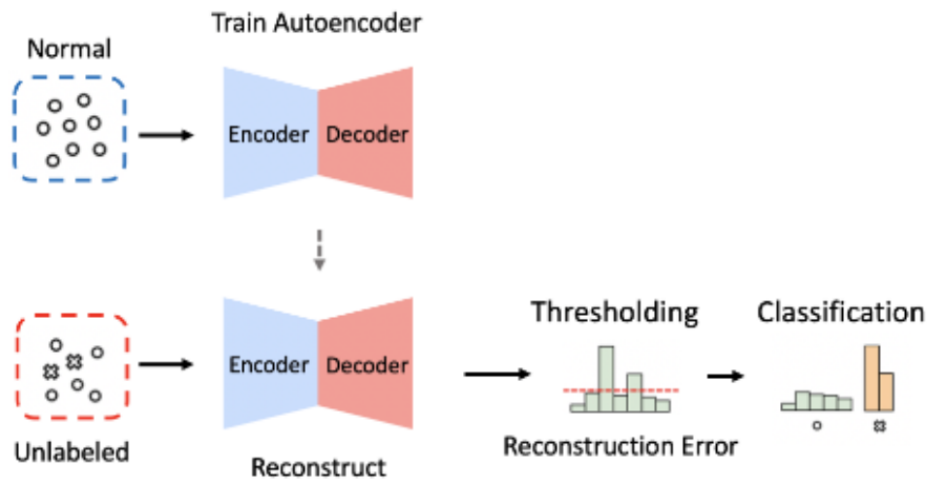


Figure 4.2 – Anomaly detection process of AutoEncoder [from Youngrok et al. (2021)]

4.3 Case study simulation data

In this section, we will test the level of performance offered by the various algorithms chosen, with or without adaptation, by applying them to an initial database of simulated data. This will be a highly imbalanced database, where abnormal data, corresponding to the opening of the window, will be rare. The proposed scenario is not necessarily realistic, but is nevertheless designed to push the algorithms to some of their limits. A second,

more realistic database will be proposed in section 4.4 with the aim of using the initial observations made in this section to define more definitively the comparative performance levels of the selected algorithms.

4.3.1 First case study – Imbalanced dataset

4.3.1.1 A first glance at the imbalanced dataset

In Chapter 2, we introduced the simulated dataset, which originally comprised thirteen distinct features. These features are expected to have a higher correlation with window opening and occupants' presence. The generated dataset is highly imbalanced regarding windows opening status. Indeed, out of 35,040 simulated time steps, only 147 instances represent window 'opening' events, resulting in a window opening rate of approximately 0.0042 for the entire dataset.

For a more targeted efficient analysis, we narrowed our focus to seven of the most relevant features from the original thirteen. These selected features include one categorical variable, the window status, and six continuous variables: computers heat gains, occupants' heat gain, solar gain from the window, heat consumption, and both indoor and outdoor temperatures.

In order to visualize the dataset into 2D or 3D representations, preserving both global and local data structures, we employ the dimensionality reduction technique known as UMAP. UMAP follows the idea that high-dimensional data can be accurately represented in lower dimensions while respecting the data's geometric characteristics. Consequently, it generates new dimensions that are combinations of the original features. These new dimensions aim to maximize the separation between different data groups, although their direct interpretation in terms of the original feature space may be less intuitive. Nonetheless, the relative distances between data points along these UMAP-generated axes hold meaning. Data points closer to each other in this space share greater similarity based on the original high-dimensional data, whereas those situated farther apart show reduced similarity. Figure 4.3 provides a 2D visualization of our input features. From this figure, it becomes apparent that the dataset is imbalanced between the closed / normal (green dots) and open / abnormal (red dots) status, with overlapping classes. The two classes are not distinctly separable based on these features. In such cases, standard clustering methods may not perform optimally.

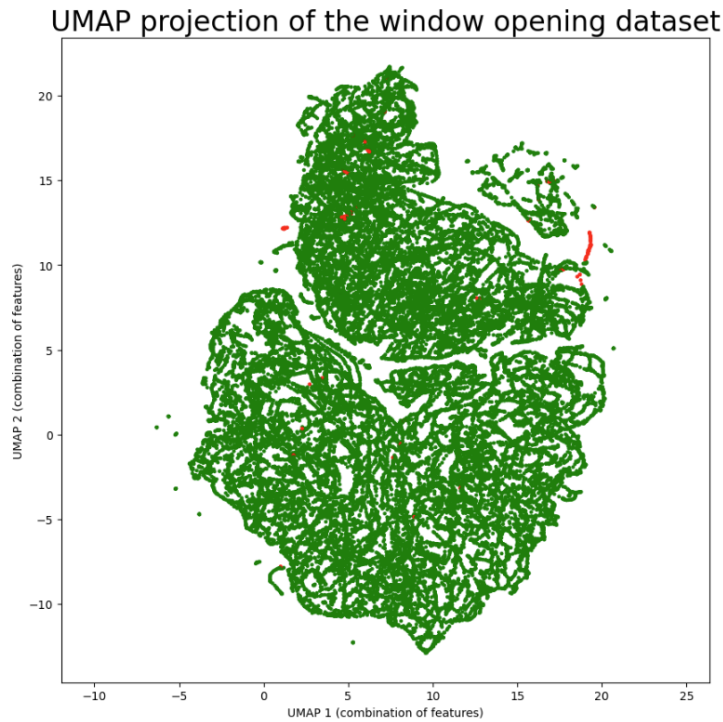


Figure 4.3 – UMAP of simulation dataset: green (closed/normal) and red (open/abnormal) dots show data imbalance

In the remainder of Section 4.3.1, we will systematically compare the various unsupervised, supervised, and semi-supervised methods according to their ability to assist in event detection, specifically window openings.

4.3.1.2 Comparison of the unsupervised methods

In our study, each unsupervised algorithm is applied to the entire year’s dataset. Six features (computer heat gains, occupants’ heat gain, solar gain from windows, heat consumption, indoor and outdoor temperatures) serve as inputs for each algorithm, with the goal of detecting anomalies in window status (seventh feature). Since this is an unsupervised approach, the window status is stored separately as the ground truth to evaluate the performance.

In our implementation, carried out using Python, we standardize all input features before applying the unsupervised methods. Standardization is achieved using the `StandardScaler` function from the `sklearn.preprocessing` module². Standardizing the features is essential as many machine learning algorithms perform optimally when input data features

2. `sklearn.preprocessing` is a module in scikit-learn that provides several common utility functions

have similar scales, approximating a standard normal distribution with a mean of 0 and a standard deviation of 1.

One of the unsupervised algorithms utilized in this study is the DBSCAN algorithm, we can use the algorithm from the `sklearn.cluster` module³. We need to specify two key parameters : Epsilon (`eps`) and `Min_samples` (see subsection 4.2.1). For this study, we randomly set epsilon to 0.1, aiming for a balance between sensitivity to outliers and the ability to cluster non-outlier data points. A common rule of thumb suggests setting `Min_samples` as ‘D+1’, where ‘D’ is the number of input features. However, this can be adjusted to suit the dataset. For this study, we set `Min_samples` to 7.

The second unsupervised learning algorithm is the PCA technique, implemented from the `sklearn.decomposition` module³ to reduce the dimensionality of our standardized feature set. Specifically, we transform our initial 6-dimensional data into 2-dimensional data for enhanced visualization and computationally efficient anomaly detection. The two dimensions chosen by PCA correspond to the directions in the original data that maximize variance, essentially representing the directions along which the data is most dispersed. After reducing the dimensionality, the original dataset is reconstructed, and the reconstruction error is calculated. Instances with a reconstruction error exceeding a predefined threshold are treated as window openings. Typically, the threshold is set as the mean plus one or two standard deviations of the reconstruction errors, following statistical conventions.

The third unsupervised learning method, MGD, we do not compute a reconstruction error as in PCA but rather calculate the likelihood of a data point within the learned multivariate Gaussian distribution. This likelihood is calculated using the probability density function (PDF) of the multivariate Gaussian distribution. MGD offers a probability distribution that assesses the likelihood of a data point under a multivariate Gaussian model. High likelihood values are associated with normal data, while anomalies yield lower likelihood scores. Therefore, a common rule of thumb for setting the threshold in MGD is to use the mean minus one or two standard deviations of the likelihoods. Instances with a likelihood below this threshold are considered anomalies. In our case study, we initially set the threshold based on this general guideline.

and transformer classes to change raw feature vectors into a representation that is more suitable for the downstream estimators.

3. it is part of the Scikit-learn library in Python

4.3.1.2.1 First comparison of the unsupervised methods without thresholds and hyper-parameters optimization

After running the window open detection models for all three algorithms, their performances are assessed using various metrics, including Precision, Recall, F1 Score, and the confusion matrix (see subsection 1.3.2.1). To facilitate visual model comparisons, we create Receiver Operating Characteristic (ROC) curves for each model. The ROC curve plots the true positive rate (Recall) against the false positive rate (1 - specificity) across various classifier thresholds. Additionally, we compute the Area Under the Curve (AUC) of the ROC curve as a single metric summarizing the classifier's overall performance.

	Precision	Recall	F1 Score
DBSCAN	0	0	0
PCA	0.497	0.042	0.077
MGD	0	0	0

Table 4.2 – Comparison of Precision, recall, and F1 Score for unsupervised DBSCAN, PCA, and MGD methods.

As a reminder from Chapter 1, let us briefly summarize the meaning of the True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) in the context of the confusion matrix:

- **True Positive (TP):** The algorithm correctly identifies a window as being open. In other words, when the ground truth is that the window is open, the algorithm also predicts it as open.
- **True Negative (TN):** The algorithm correctly identifies a window as being closed. When the ground truth is that the window is closed, the algorithm also predicts it as closed.
- **False Positive (FP):** The algorithm incorrectly identifies a window as being open. This occurs when the algorithm predicts that a window is open when, in fact, it is closed. This is akin to "inventing openings" that do not exist in reality.
- **False Negative (FN):** The algorithm incorrectly identifies a window as being closed. In this case, the algorithm predicts a window as closed when it is actually open. This means the algorithm is "missing real openings."

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
DBSCAN	23609	11284	147	0
PCA	33215	1678	74	73
MGD	33288	1605	147	0

Table 4.3 – Comparison of confusion matrix results for DBSCAN, PCA, and MGD methods

In Table 4.2, along with the corresponding confusion matrix in Table 4.3, we can observe that with the initial parameters and threshold selections, all methods exhibit fair recall but not-so-great precision. This suggests that all models have a false positive rate, meaning they incorrectly identify many positive cases as negative. Specifically, both DBSCAN and MGD have a TP value of 0, indicating that they never correctly identify real openings. This implies that these models are not very precise in their predictions, and they tend to "invent" window openings.

4.3.1.2.2 Comparison of the unsupervised methods after thresholds and hyper-parameters fine tuning by grid search

It is worth noting that the selection of parameters and thresholds is crucial for the performance of unsupervised models. Sometimes, there's a trade-off involved because setting the threshold too low can lead to many false positives (resulting in low precision), while setting it too high can result in many false negatives (leading to low recall). Considering this, we plan to conduct a grid search for all potential parameter values for DBSCAN⁴ parameters, and PCA, MGD thresholds. The F1 score will be used as a criterion because it balances the trade-off between precision and recall. The grid search will explore various parameter combinations and thresholds to identify the ones that optimize the results.

Figure 4.4 displays the ROC curve and precision-recall curve for DBSCAN with various parameter combinations. The red line represents the best-performing DBSCAN model,

4. For ϵ (Epsilon): Lower Bound (0.1): A very small ϵ would mean that a point's neighborhood would be too limited, possibly leading to many small clusters or even making most points noise. This is often undesirable in clustering contexts.

Upper Bound (1): A very large ϵ would include too many points in the neighborhood, potentially merging clusters that should be separate. Keeping it at 1 ensures that the clusters are not overly broad. For `min_samples`: Lower Bound (1): A smaller `min_samples` value would make it easier for points to become core points, thereby potentially creating more clusters, some of which might just be noise. A value of 1 is a reasonable starting point that balances sensitivity and specificity.

Upper Bound (20): A larger `min_samples` value would make it harder to form clusters, as more neighboring points would be needed for a point to become a core point. This could result in many points being labeled as noise. The upper bound of 20 is chosen.

which achieved the highest F1 score. Figure 4.5 illustrates how the F1 score varies with different threshold selections for PCA and MGD.

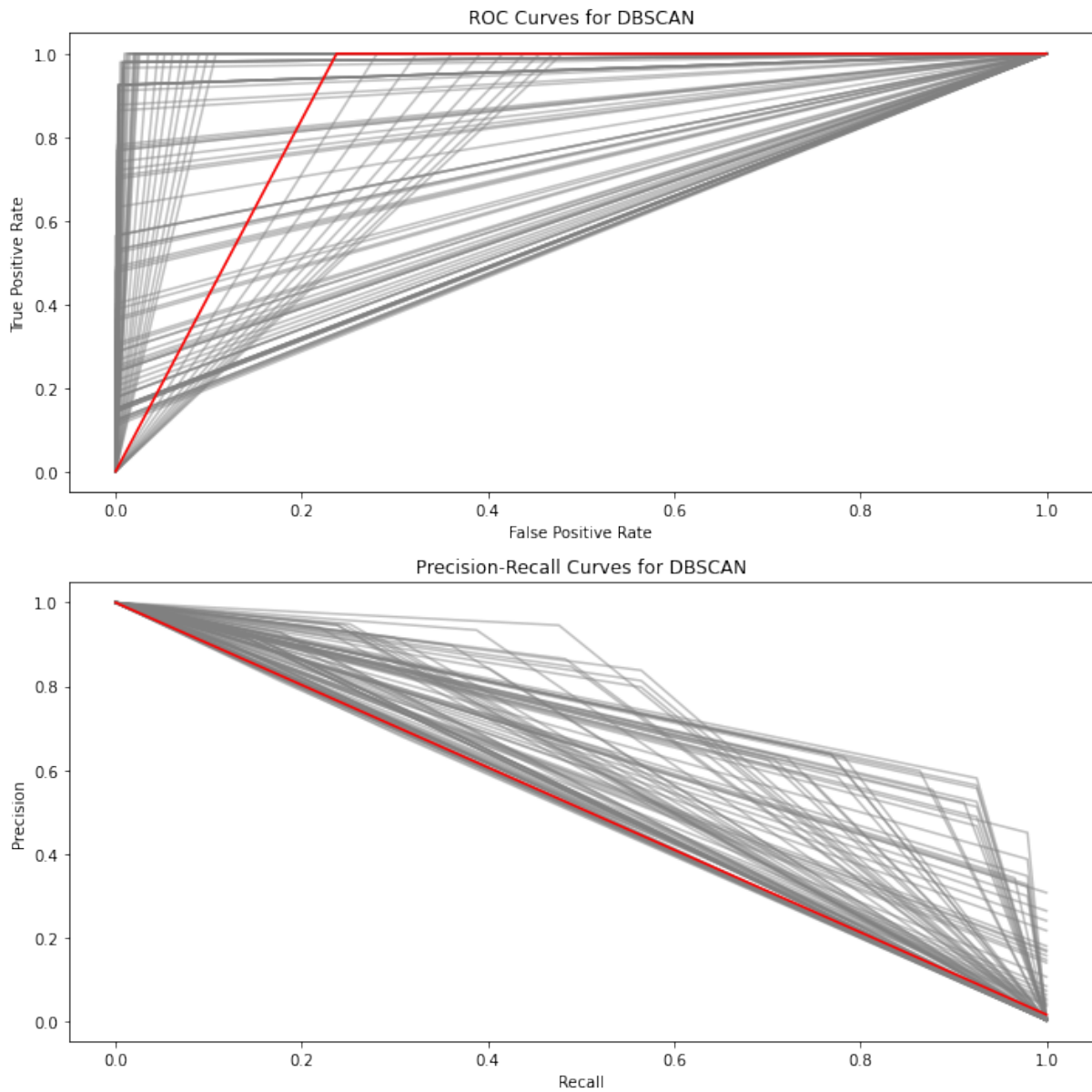


Figure 4.4 – Grid search for DBSCAN parameters chosen

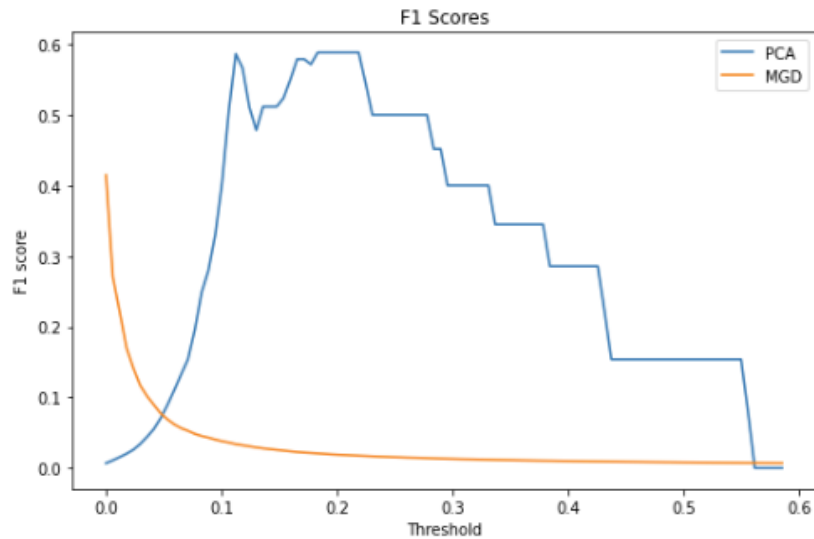


Figure 4.5 – Grid search for PCA and MGD threshold chosen

Table 4.4 provides a comparison between the original and optimized values for thresholds and hyperparameters. In the case of DBSCAN, a significant deviation in the best epsilon value from the initial setting may suggest that the original value did not adequately capture the inherent density variations among the clusters. Similarly, a substantial change in the optimized `Min_samples` value compared to the initial one could indicate that the initial setting was either too restrictive or too permissive for cluster formation.

For PCA, a higher new threshold compared to the original suggests that the initial setting was likely too conservative, potentially flagging an excessive number of instances as anomalies. Conversely, a lower threshold would suggest the opposite.

In the context of MGD, a markedly different new threshold from the original could imply that the initial concept of "normality" was inaccurate. A higher threshold would mean that a larger number of points are now considered "normal," whereas a lower one would imply a stricter criterion for "normality."

Clearly, the selection of parameters and thresholds has a significant impact on the analysis outcomes. However, crafting a universal rule for this purpose is challenging due to the diversity of data characteristics and specific problem requirements. Furthermore, in real-world scenarios, ground truth data is often unavailable, complicating the validation process.

Method	New Threshold or parameters	Original Threshold or parameters
DBSCAN	eps=0.5, min_samples=10	eps=0.25, min_samples=7
PCA	1.83e-01	8.15e-01
MGD	5.22e-10	1.15e-04

Table 4.4 – Comparison of Original and New Thresholds and Parameters for DBSCAN, PCA, and MGD

Table 4.5 shows the new precision, recall, and F1 score after finding the best parameters and threshold, we can notice that all F1 scores have a significant improvement.

	Precision	Recall	F1 Score
DBSCAN	0.925	0.581	0.714
PCA	0.442	0.929	0.599
MGD	0.830	0.348	0.490

Table 4.5 – Comparison of precision, recall, and F1 Score for DBSCAN, PCA, and MGD methods after Grid search

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)	Time (seconds)
DBSCAN	34795	98	11	136	6.007
PCA	34888	5	82	65	0.309
MGD	34664	229	25	122	0.014

Table 4.6 – Comparison of confusion matrix results and computation times for DBSCAN, PCA, and MGD methods after Grid search

From the results, DBSCAN exhibits the highest precision (0.925) and F1 score (0.714) among the three methods. Furthermore, in Figure 4.6, we can observe the ROC curve, where the area under the ROC curve, often referred to as the AUC (Area Under the Curve), provides a measure of how well a parameter can distinguish between two diagnostic groups (abnormal/normal). The higher the AUC, the better the model generally performs. This is because the curve will approach to the top-left corner of the plot, which corresponds to a greater true positive rate and a lower false positive rate.

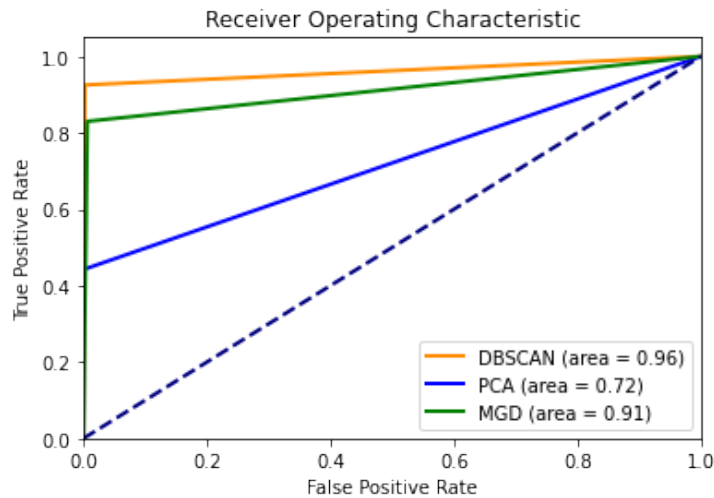


Figure 4.6 – ROC curve for three algorithms

DBSCAN shows an AUC of 0.96, indicating that the model has a good trade-off between sensitivity (true positive rate) and specificity (1 - false positive rate). In simple terms, the model can correctly identify a large proportion of positive cases (high True positive rate) while maintaining a relatively low number of false alarms (False positive rate). The high recall and ROC area indicate that DBSCAN is very effective at identifying positive cases (anomalies) in the data. However, the precision is relatively low (0.581), implying that it might classify instances as anomalies somewhat generously, leading to a higher rate of false positives. The density-based nature of DBSCAN could account for its high recall and ROC area, as it does not assume a specific data distribution, and it is capable of discovering clusters of various shapes and sizes.

PCA shows the highest precision (0.928) among the three methods. This means that when PCA predicts an anomaly, it is very likely to be correct. However, the recall is relatively low (0.442), suggesting that PCA misses a significant number of anomalies. The ROC area is 0.72, which is lower than that of DBSCAN and MGD, indicating a less optimal trade-off between sensitivity and specificity. These results for PCA might be attributed to its assumptions and nature. PCA assumes linear correlations among features and tries to capture the majority of the data's variance in fewer dimensions. However, if anomalies do not follow these linear patterns, PCA might struggle to detect them, leading to a lower recall.

The MGD method has the lowest precision (0.347) but a relatively high recall (0.829). The F1 score is also the lowest among the three (0.489). However, the ROC area is 0.91,

which is higher than PCA but lower than DBSCAN. This suggests that despite its lower precision and F1 score, MGD may provide a better trade-off between sensitivity and specificity than PCA. The results for MGD could be explained by its nature. Indeed, MGD assumes a Gaussian distribution for the data, and its effectiveness heavily depends on how well this assumption holds. It is also sensitive to the initial parameters and the number of iterations, which could impact its performance.

In term of computation time, DBSCAN took the longest time to execute, with a computation time of 6.007 seconds. This is significantly higher compared to the other two algorithms. Given that it is a density-based clustering algorithm, it might take longer on certain datasets, especially if there are many data points to cluster. PCA, being a dimensionality reduction technique, generally has a moderate computation time, especially when dealing with high-dimensional data. However, in this context, it is much quicker than DBSCAN. MGD was the fastest among the three, with a computation time of just 0.014 seconds. This suggests that MGD is highly efficient in terms of time complexity, at least for the dataset in question.

To sum up, DBSCAN performs best in terms of precision, F1 score, and ROC area, indicating its effectiveness in anomaly detection in this case. However, the best method depends on the specific needs of task. If precision is of utmost important (minimizing false positives), then DBSCAN would be the preferred choice. Conversely, if maximizing recall (capturing as many anomalies as possible) is the goal, then PCA is the best option. If seeking a balance between these metrics, we might favor DBSCAN.

4.3.1.2.3 Comparison of the unsupervised methods after thresholds and hyper-parameters optimization

In unsupervised method, the absence of ground truth labels makes the trial-and-error approach ineffective for fine-tuning each algorithm's parameters and threshold. In such cases, visualization methods become valuable assistants for parameter selection. In the case of DBSCAN, we can use the k-distance graph as a tool for determining the parameters in DBSCAN, especially the epsilon parameter. The k-distance of a data point refers to its distance to the k-th nearest neighbor. Calculating this metric for all points allows us to understand the distribution of local densities. When these k-distances are plotted

in ascending order, we gain valuable insights. Specifically, a data point located within a dense cluster will have its k -th nearest neighbor close by, resulting in a small k -distance. Conversely, a point in a less dense area will exhibit a larger k -distance. When we create a k -distance plot, we look for the 'elbow' point, where the graph sharply turns, indicating a significant increase in k -distances. This 'elbow' point corresponds to the distance at which points are no longer densely clustered, effectively distinguishing dense regions (potential clusters) from sparse regions (potential noise). Setting epsilon to the k -distance at the 'elbow' point ensures that DBSCAN forms clusters with points in dense regions while classifying points in sparse regions as noise.

The overall process can be summarized as follows:

1. **Compute the k -distances for all points:** Calculate the distance to the k -th nearest neighbor for each point in the dataset. The value of k is typically chosen by the user, often set to a low value like $k = 4$.
2. **Sort and plot the k -distances:** Sort the computed k -distances in ascending order, and create a plot. The y-axis represents the k -distance, while the x-axis displays the sorted points.
3. **Identify the 'eps' parameter from the graph:** Determine the 'eps' parameter by locating the y-coordinate of the 'elbow' point in the k -distance graph. By choosing this point as 'eps', we can cover as many points in the same neighborhood as possible, while excluding those that are farther away, likely belonging to other clusters or representing noise.

In Figure 4.7, we observe the "elbow" point, and the corresponding epsilon value is approximately 0.5, which aligns with our previous grid search results.

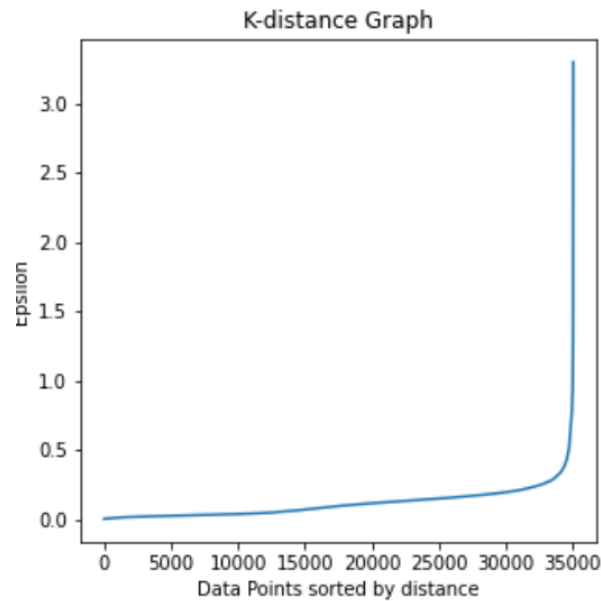


Figure 4.7 – K-distance graph

For PCA-based anomaly detection, visualization methods can be useful in determining the threshold for classifying a data point as an anomaly. A common approach involves plotting the reconstruction error against instances and determining a value that keeps most reconstruction below a certain threshold. Reconstruction error is computed as the difference between the original data and the reconstructed data (i.e., after transforming to the PCA space and then inversely transforming back to the original space). In the resulting Figure 4.8, an appropriate threshold might be a value close to 0 since most values have a small reconstruction error. To sum up, because it relies on a visualization-based method, the threshold chosen is all about the balance between precision and recall that we wish to achieve. A low threshold increases recall but might reduce precision. It is often beneficial to experiment with different thresholds to determine the most suitable one for a specific use case.

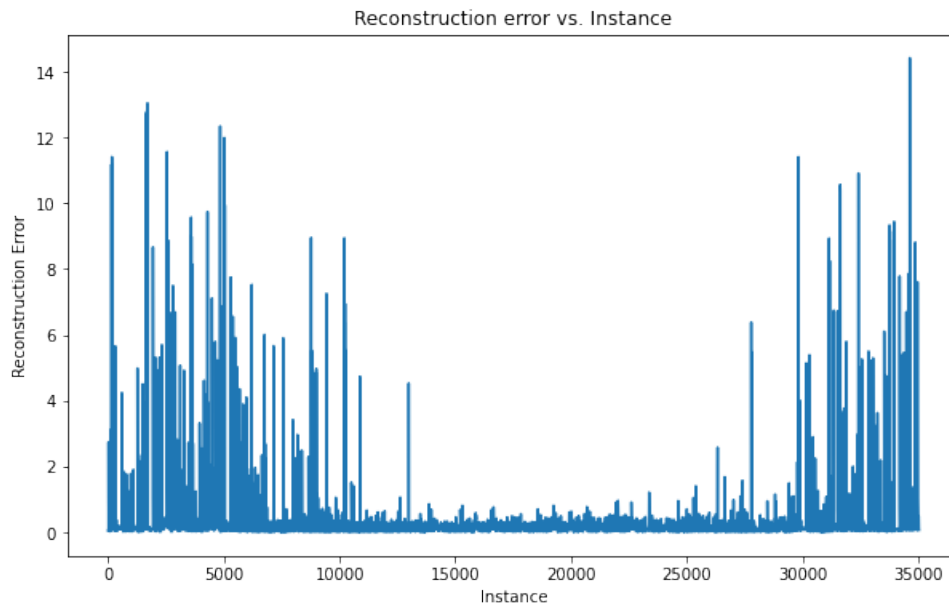


Figure 4.8 – PCA reconstruction error VS instance

For MGD-based anomaly detection, a common way to visualize and determine the threshold is by using a histogram of the calculated probabilities (or scores) for instances in the training set. The idea is to compute the probability of each instance under the learned multivariate Gaussian distribution, and then plot a histogram of these probabilities. Typically, normal instances should have higher probabilities, while anomalies should have lower probabilities. In the resulting Figure 4.9, the histogram displays a right-skewed distribution, indicating that most data points are considered as "normal", with only a few classified as anomalies by the MGD model. The threshold should be set after the 0.05 percentile, and comparing it with the value from the grid search, this choice should yield good precision but might affect recall. As with other anomaly detection techniques, selecting an optimal threshold often involves some trial and error.

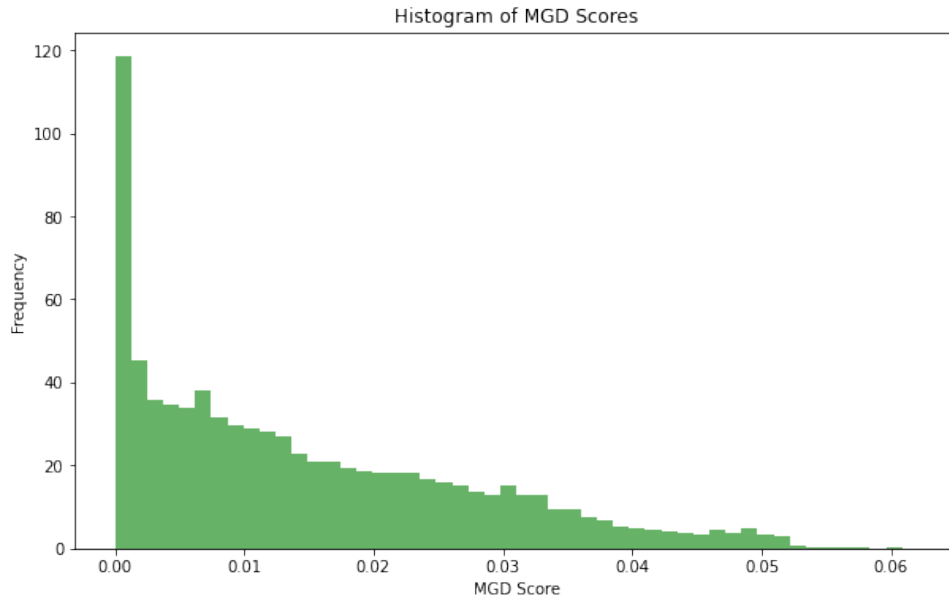


Figure 4.9 – Histogram of MGD scores

To conclude with this subsection dedicated to the comparison of the unsupervised learning algorithms selected, DBSCAN exhibits the highest number of True Positives (136) and the lowest number of False Positives (11). This suggests that it excels in correctly identifying openings while maintaining a low rate of false alarms. PCA has the highest number of True Negatives (34888) and the lowest number of False Negatives (5), indicating it is excellent at identifying non-openings and not missing actual openings. MGD has a higher number of False Negatives (229), meaning it misses quite a few actual openings.

4.3.1.3 Comparison of the supervised methods

4.3.1.3.1 Imbalance data issue for supervised methods

As discussed in Chapter 1, dealing with class imbalance is a significant challenge in supervised activity detection. Learning effective rules from a supervised model with a limited number of negative samples (outliers) becomes more challenging when facing strong class imbalance, which is present in our study due to the very low window opening rate of approximately 0.0042 in the entire dataset.

In the following sections, we will explore two preprocessing methods to tackle class imbalance and compared their efficiency in terms of improvement compared to the results obtained without addressing the class imbalance.

On one hand, we perform an undersampling (Liu et al. 2009), which involves selecting a small number of samples at random from the majority class (in our study, the closed window state class). These selected samples are then combined with the original minority class samples to create a new training dataset. On the other hand oversampling, where the minority class is expanded by duplicating observations from that class is used. The random oversampling algorithm is the simplest approach, but it can lead to model overfitting as it makes the learned information too specific. To mitigate this, data synthesis techniques, such as the Synthetic Minority Oversampling Technique (SMOTE) (Chawla et al. 2002), are used in our study to generate more data based on existing data.

4.3.1.3.2 Data preprocessing and splitting

In supervised methods for activity detection, the dataset undergoes preprocessing before being divided into training and testing subsets. For data preprocessing, the dataset, encompassing one year’s worth of data from six features (computers’ heat gains, occupants’ heat gain, solar gain from windows, heat consumption, and indoor and outdoor temperatures), is loaded into a pandas DataFrame⁵. These features serve as inputs for each supervised algorithm, with window status (open or closed) as the output variable. To ensure uniformity, the data are normalized using `StandardScaler` from `sklearn.preprocessing`, which scales the features to have a zero mean and unit variance. Subsequently, the preprocessed dataset is then partitioned into training and testing sets in a 80:20 ratio. This split is achieved with the help of `train_test_split` module from `sklearn.model_selection`⁶. Under this configuration, 80% of the data is allocated for training the model, while the remaining 20% is reserved for testing its performance.

4.3.1.3.3 Model training and Evaluation with Logistic Regression

As far as the `LogisticRegression` model is concerned, it is instantiated using the `sklearn.linear_model`⁷ library and then fitted to the training data. Once trained, the model is applied to the unseen test data for making predictions. Subsequently, the model’s performance similar as the following XGBoost and ANN methods, is evaluated using evaluation metrics such as Precision, Recall, and F1 Score, all of which are available in

5. Pandas is a Python package used for data manipulation and analysis.

6. Scikit-learn (`sklearn`) is a machine learning library for Python.

7. The `sklearn.linear` model module in scikit-learn provides tools for logistic regression.

the `sklearn.metrics`⁸ library.

For XGBoost classifier, it is initiated using the `xgboost.XGBClassifier`⁹ library. Several critical hyperparameters are configured to optimize the model's performance. The `learning_rate` is set to 0.1. This small learning rate ensures the model's stability by taking smaller optimization steps during training. To prevent overfitting, the `max_depth` is restricted to 3. Limiting the depth of the trees is a common practice in boosting frameworks, favoring smaller trees to maintain weak individual learners. The `n_estimators` hyperparameter is set to 100. This parameter controls the number of boosting rounds and ultimately determines how many decision trees will be constructed. A value of 100 strikes a balance between model complexity and computation time. Regularization parameters, `lambda` and `alpha`, are set to 1. These correspond to L2 and L1 regularization terms, respectively. Regularization plays a crucial role in controlling overfitting by introducing penalties to the model's complexity. Setting them to 1 represents a moderate choice that helps in mitigating overfitting without excessively penalizing the model. The model employs the default "binary:logistic" objective function for binary classification. This objective function combines both the loss function and regularization terms, providing a comprehensive approach to the classification task. Training the model begins with the `fit` method applied to the training dataset, which includes feature variables and the target variable, indicating whether the window is open or closed. During training, decision trees are constructed sequentially, with each tree designed to correct the errors of the cumulative preceding trees. The tree-building process involves identifying optimal splits based on criteria like Gini impurity and information gain. It calculates gradient and Hessian values to determine these optimal splits and optimizes leaf nodes to minimize the loss function. After constructing each tree, the algorithm prunes leaves and branches that do not contribute sufficient gain, following guidelines set by the regularization parameters (`gamma`). Fine-tuning of hyperparameters such as `learning_rate`, `max_depth`, and `n_estimators` can be performed using cross-validation techniques like k-fold cross-validation, which is available through `xgboost.cv`.

Finally, for training and evaluating ANN, we follow a structured process. First, concerning the model architecture, `tensorflow.keras.models`¹⁰ library is used to build a sequential model. The architecture of the model involves an input layer with twelve nodes

8. `sklearn.metrics` provides various metrics for evaluating machine learning models.

9. XGBoost is an open-source software library that provides gradient boosting for Python.

10. TensorFlow is an open-source machine learning framework, and Keras is its high-level API for building neural networks.

and two hidden layers with twelve and eight nodes, respectively. These node configurations are chosen to allow the network to capture sufficient complexity without risking overfitting. The final layer consists of one output node. We apply the 'relu' activation function in the input and hidden layers, which is known for handling complex patterns effectively. For the output layer, we use the 'sigmoid' activation function. The ANN structure is shown in Figure 4.10.

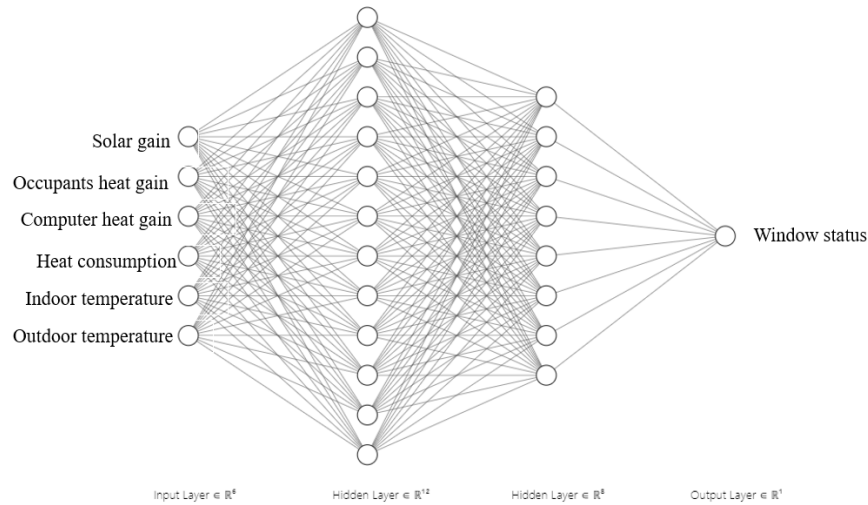


Figure 4.10 – ANN structure

After designing the architecture, the model is compiled. In our case, we use binary cross-entropy as the loss function, the Adam optimization algorithm¹¹ for training, and accuracy as the metric to measure the model's performance. Next, we train the ANN on the training dataset. We choose to train it for 150 epochs, where each epoch represents a complete pass through the entire training dataset. It is important to strike a balance here, as training for too many epochs can lead to overfitting. Additionally, we use a batch size of 10, which determines the number of samples used in each weight update during training. Smaller batch sizes introduce regularization and reduce generalization error, but excessively small batches can result in unstable convergence. Our choice of 150 epochs indicates that it was sufficient for the model to learn the data without significant overfitting. A batch size of 10 strikes a balance between computational efficiency and

11. Adam is a popular optimization algorithm for training deep learning models, well-known for its efficiency and low memory requirements. It mixed the advantages of two other stochastic gradient descent extensions: AdaGrad, effective with sparse gradients, and RMSProp, well-suited for online and non-stationary settings.

model performance.

4.3.1.3.4 Supervised methods - Effect of (re)sampling

In Figure 4.11 and 4.13, we present the learning curve (LC) respectively for logistic regression, XGBoost and ANN, without sampling (i.e., with the original dataset). Learning curves offer insights into the model’s behavior and learning process, often consisting of two parts: the training learning curve and the validation learning curve. The training learning curve reflects how well the model learns from the training data. A descending curve indicates effective learning, while a flat curve suggests overfitting or a lack of learning. The validation learning curve displays the model’s performance on an unseen dataset, commonly referred to as the validation set, providing important insights into its generalization capabilities. When both curves reach a plateau, it indicates underfitting. If the validation curve rises while the training curve continues to decrease, it suggests overfitting. However, if both curves converge to a point with minimal errors and a small gap between them, it signifies a good fit.

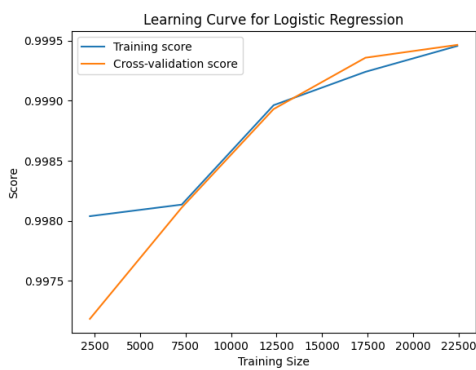


Figure 4.11 – LC logistic regression

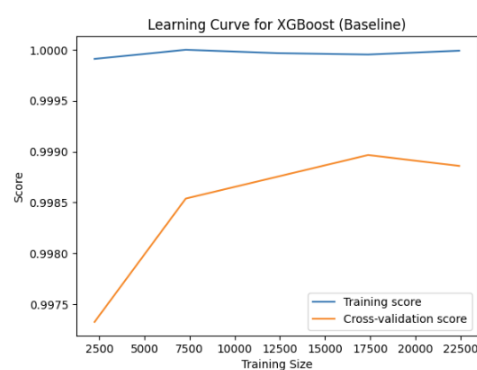


Figure 4.12 – LC XGBoost

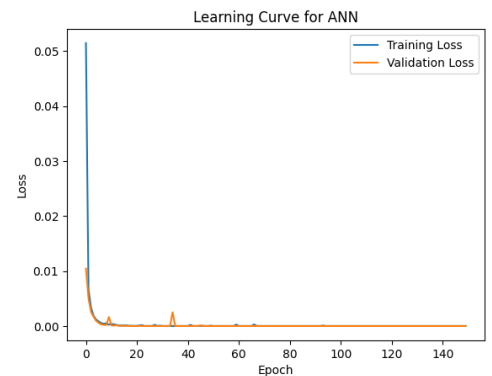


Figure 4.13 – LC ANN

For logistic regression, both training and validation scores continue to rise as the training dataset size increases. This suggests that the model is still learning and could benefit from additional data. Logistic regression attains a balanced accuracy of 0.875, meaning that the model correctly classifies 87.5% of instances for both the positive and negative classes, considering the dataset’s imbalance.

Effect of Undersampling

After undersampling in logistic regression, the Balanced Accuracy and ROC AUC Score increase to 0.97 and 0.99, respectively. This indicates improved overall performance across both classes and stronger class separability. Undersampling does not significantly improve XGBoost and ANN, and even leads to a significant reduction in the precision and F1 score of XGBoost. This is because ANNs, especially deep networks, and advanced boosting algorithms like XGBoost are highly complex models capable of fitting very complex boundaries. These models might already be performing well on imbalanced data by capturing the underlying complexities of the data, so undersampling doesn't lead to a noticeable performance improvement. And undersampling involves removing instances from the majority class, which can result in a significant loss of information. If the instances removed contain important nuances or patterns, the model might perform worse because it's not seeing the full picture of the data.

Figures 4.14 to 4.16 focus on the specific case of logistic regression where learning curves with different sampling methods are compared. From the learning curve in Figure 4.14, at the beginning of training, the model quickly learns to correctly classify the majority of instances, resulting in a rapid increase in the cross-validation score. Over time, as the model begins to fit the training data more closely, both training and cross-validation scores gradually increase. Overall, the model has achieved a good fit after undersampling without overfitting and underfitting. But even though the recall has increased, the precision is dropped significantly, because undersampling discards information from the majority class to make the dataset balanced. In this case, the oversampling or SMOTE should be considered.

Effect of Oversampling and SMOTE

Following oversampling and SMOTE, logistic regression achieves perfect results, as demonstrated in Table 4.7. This improvement results from the increased representation of the minority class, enabling the model to better understand the characteristics of these instances and make more accurate predictions. However, examining the learning curves (refer to Figures 4.15 and 4.16) raises concerns about overfitting. The training score remains consistently high, while the cross-validation score significantly decreases after a certain point. This pattern suggests that the model performs exceptionally well on the training data but struggles to generalize to unseen data.

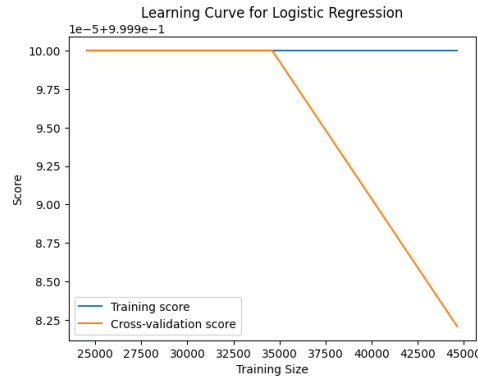
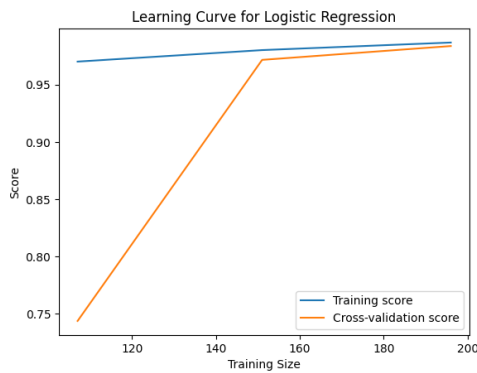


Figure 4.14 – Logistic regression Undersampling

Figure 4.15 – Logistic regression Oversampling

Figure 4.16 – Logistic regression SMOTE

To address potential overfitting in logistic regression, we employ cross-validation. In this approach, the original dataset is randomly divided into five equal-sized subsamples. One subsample serves as the validation data for testing the model, while the remaining four subsamples constitute the training data. This cross-validation process repeats four times, with each subsample acting as the validation data once. Figure 4.17 provides an illustration of this process. By validating the model on a hold-out set not used during training, we can evaluate how well it is likely to perform on new data. If the model excels on the training data but performs poorly on the validation data, it is indicative of overfitting, implying that the model does not generalize effectively to new data.

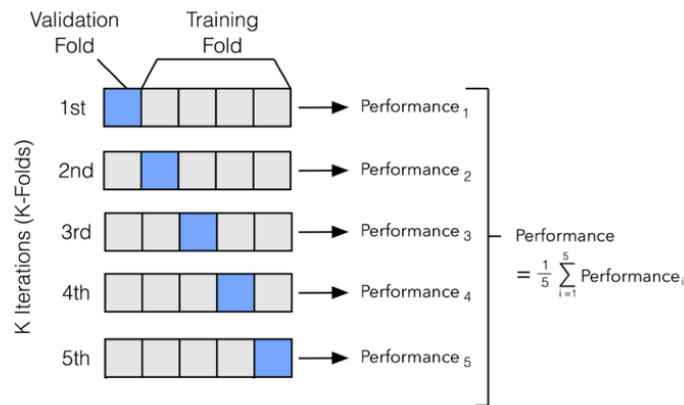


Figure 4.17 – 5 folder cross validation

Furthermore, employing multiple rounds of cross-validation with varying data subsets in each round contributes to a more robust performance estimate. This approach mitigates

the risk of overly optimistic estimates resulting from a fortunate choice of train/test splits. Consequently, it reduces the likelihood of selecting an overly complex model that merely performs well on a particular data partition. The Violin plot¹² depicted in Figure 4.18 demonstrates the effectiveness of cross-validation in both oversampling and SMOTE. Moreover, it indicates the absence of overfitting in both of these algorithms.

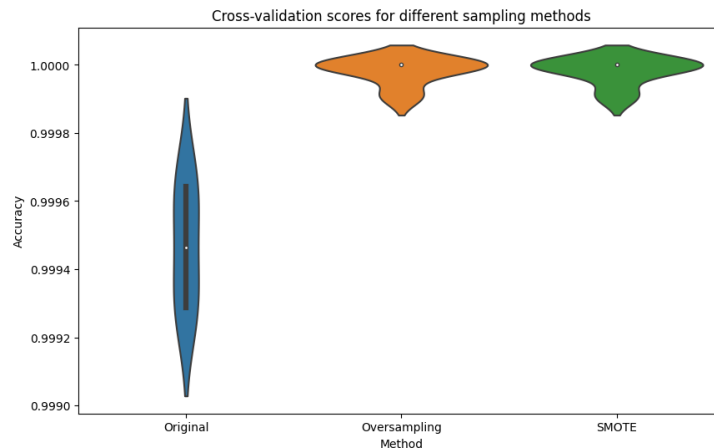


Figure 4.18 – Violin plot

The learning curve for XGBoost in Figure 4.12 indicates overfitting, a common issue, particularly with highly imbalanced data. XGBoost has learned the training data thoroughly but fails to generalize to new data. However, with data preprocessing using SMOTE and Oversampling, XGBoost starts to achieve perfect results, with both balanced accuracy and ROC AUC score reaching 1, as shown in Tables 4.7 and 4.8.

For the ANN (Figure 4.13), the training and validation losses decrease to zero, and precision, recall, and F1 scores confirm this "perfection" as shown in the Table 4.7 (row "ANN (Original)"). This "perfection" possibly indicated overfitting. To dispel this doubt, we need additional information such as balanced accuracy¹³, AUC-ROC¹⁴ that account for imbalanced datasets. Table 4.8 compiles these other metrics which score a perfect 1.0 for ANN with original dataset. It confirms the exceptional modelling performance of ANN.

12. In a Violin plot, the width of the violin at a given accuracy level (y-coordinate) indicates how many times that accuracy level was observed across the five folds (model frequently achieved that accuracy score), and a narrower part means that the model less frequently achieved that accuracy level

13. The Balanced Accuracy Score is an average of recall (or sensitivity) obtained on each class. It's a useful metric when dealing with imbalanced datasets.

14. AUC-ROC score is the area under the receiver operating characteristic (ROC) curve, it is usually computed using trapezoidal rule integration over the ROC curve

	Precision	Recall	F1 Score	Computation Time (s)
Logistic Regression (Original)	1.0	0.75	0.86	0.14
Logistic Regression (Undersampling)	0.32	1.0	0.48	0.1
Logistic Regression (Oversampling)	1.0	1.0	1.0	0.11
Logistic Regression (SMOTE)	1.0	1.0	1.0	0.12
XGBoost (Original)	1.0	0.8966	0.9455	3.05
XGBoost (UnderSampling)	0.0900	0.9655	0.1647	0.05
XGBoost (OverSampling)	1.0	0.9655	0.9825	2.48
XGBoost (SMOTE)	1.0	1.0	1.0	3.66
ANN (Original)	1.0	1.0	1.0	924
ANN (Undersampling)	1.0	1.0	1.0	202
ANN (Oversampling)	1.0	1.0	1.0	1582
ANN (SMOTE)	1.0	1.0	1.0	1642

Table 4.7 – Performance Metrics for Different Models and Sampling Techniques

Method	Balanced Accuracy	ROC AUC Score
Logistic Regression (Original)	0.875	1.0
Logistic Regression (Undersampling)	0.9766	0.9995
Logistic Regression (Oversampling)	1.0	1.0
Logistic Regression (SMOTE)	1.0	1.0
XGBoost (Original)	0.9483	0.9483
XGBoost (UnderSampling)	0.9625	0.9625
XGBoost (OverSampling)	0.9828	0.9828
XGBoost (SMOTE)	1.0	1.0
ANN (Original)	1.0	1.0
ANN (Undersampling)	1.0	1.0
ANN (Oversampling)	1.0	1.0
ANN (SMOTE)	1.0	1.0

Table 4.8 – Balanced Accuracy and ROC AUC Score for Different Models and Sampling Techniques

To conclude this section, XGBoost and ANN consistently outperformed or achieved equal scores compared to Logistic Regression. This superiority could be attributed to ANNs' ability to model intricate data relationships. However, it is important to note that ANNs come with significantly higher computational complexity, as reflected in Table 4.7, where the ANN consumed more time and resources. Decision trees, on the other hand, are simpler and offer greater interpretability compared to neural networks. They require fewer computations and iterations to capture complex patterns. XGBoost, in particular, excels in handling sparse and categorical data efficiently, and it can deal with parallel and distributed computing more effectively than ANNs, leading to faster results. Given its strong performance, XGBoost is a favorable choice for single or multi-activity detection tasks. Meanwhile, logistic regression, when coupled with oversampling and SMOTE, also delivers promising results. Further analysis will be needed for datasets with greater complexity.

4.3.1.4 Semi-supervised method result

The AutoEncoder model utilized in this first simulated data case study comprises an input layer, an encoder, a decoder, and an output layer (see Figure 4.19). The choice of a single encoder and decoder design prioritizes model simplicity and efficiency. While adding more layers to the encoder and decoder could enhance the model's capacity to learn intricate representations, it also elevates the risk of overfitting and necessitates increased computational resources.

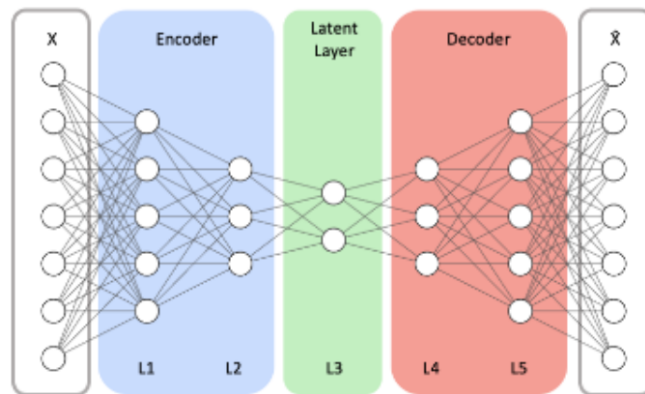


Figure 4.19 – AutoEncoder architecture

The input layer's dimensionality of our AutoEncoder aligns with the number of features in the dataset. The encoder, a feedforward network, reduces the input data's dimensionality to learn a compact representation. In our study, the encoder consists of two layers, with the first layer comprising fourteen neurons and the second containing half that number. The LeakyReLU activation function, mitigating the "dying ReLU" problem by allowing small negative values for inputs less than zero, is employed. This makes LeakyReLU suitable for preserving gradient flow.

The decoder mirrors the encoder's architecture. It takes the compressed data from the encoder and reconstructs the original input data. Similar to the encoder, the decoder comprises two layers with seven and fourteen neurons, respectively, and utilizes the LeakyReLU activation function. The output layer generates the final reconstruction of the input data.

The AutoEncoder's objective is to minimize the discrepancy between input and output, which is the reconstruction error. This is accomplished by employing mean squared error as the loss function and the Adam optimization algorithm. In the semi-supervised learning

context, the data is divided into an 80% training set and a 20% test set to assess model performance on unseen data. During training, the AutoEncoder learns to reconstruct only normal instances (window closed) with low error. Anomalies (window open), being dissimilar to what the model has learned, result in higher reconstruction errors. For each instance, the mean squared error (MSE) between the original and reconstructed data is computed. Instances with an MSE exceeding a predefined threshold are deemed anomalies. The threshold is set to the mean MSE plus 2.5 times the standard deviation of the MSE on the training set, accommodating the variation in the reconstruction error. This statistical technique is commonly used to detect outliers. The model’s performance is finally evaluated using the classical precision, recall, and F1 score metrics (see Table 4.10), computed from the model’s predictions’ confusion matrix (see Table 4.9).

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
AutoEncoder	6983	1	11	13

Table 4.9 – Confusion matrix for AutoEncoder

Metric	Score
Precision	0.542
Recall	0.929
F1 Score	0.684

Table 4.10 – Performance Metrics for AutoEncoder

The results in Table 4.9 reveal that the AutoEncoder model exhibits a high number of False Positives (FP=11) in comparison to True Positives (TP=13). This suggests that the model tends to make incorrect predictions of the window being open when it is, in fact, closed. While its precision, recall, and F1 score performance (Table 4.10) falls short of that achieved by supervised learning, it outperforms unsupervised learning methods without the selection of an optimal threshold. Since the AutoEncoder’s performance hinges on threshold selection, a grid search similar to that conducted for unsupervised methods was employed to determine the threshold yielding the highest F1 score. Figure 4.20 illustrates the F1 score variation with changing thresholds, along with the updated confusion matrix. The refined precision, recall, and F1 score metrics underscore the AutoEncoder’s enhanced performance in detecting window openings. Notably, the optimal threshold improved the model’s performance, particularly by eliminating false negatives and increasing true pos-

itives. Consequently, the model does not miss any genuine instances of window closure, affirming the AutoEncoder’s mastery of this feature. The model achieves a perfect recall score of 1.0, indicating that it identifies every actual window opening correctly. With a precision score of 0.625, the model accurately classifies 62.5% of the instances it predicts as window closures. The F1 score, representing the harmonic mean of precision and recall, reaches 0.77. While this result does not match the performance of supervised learning methods, it aligns with the levels achieved by unsupervised methods such as DBSCAN, which also yielded an F1 score of 0.71.

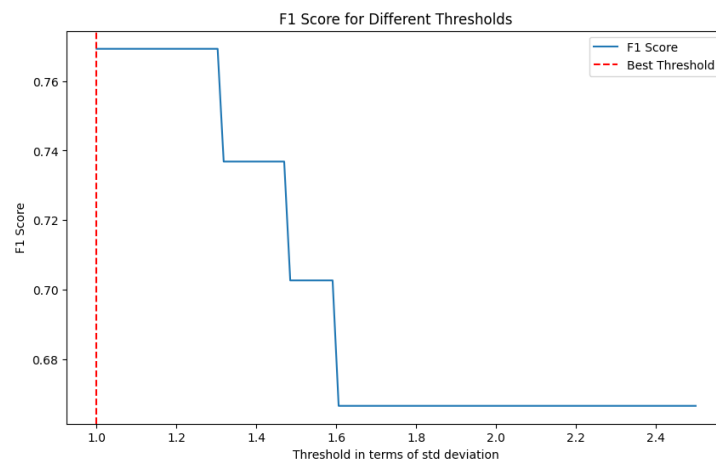


Figure 4.20 – Finding the best threshold for AutoEncoder

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
AutoEncoder	6984	0	9	15

Table 4.11 – Confusion matrix for AutoEncoder with optimal threshold

Metric	Score
Precision	0.625
Recall	1
F1 Score	0.77

Table 4.12 – Performance Metrics for AutoEncoder with optimal threshold

4.3.2 Second case study – A more realistic and balanced dataset

4.3.2.1 New simulation dataset

The methods employed in the previous sections were applied to raw data collected over a single year, and the scenarios considered for window opening were deemed unrealistic, resulting in a significant class imbalance. This unfavorable situation limited the algorithms' performance potential. In the subsequent sub-sections, we aim to compare the performance of the algorithms on a new (more) balanced, and thus realistic dataset and to study ways of improvement. The new simulation is conducted using DesignBuilder software, spanning one-year simulation with 5-minute intervals. Several assumptions were made regarding new occupancy and window schedules for this simulation. Occupancy is considered to occur on weekdays from 9 am to 7 pm, with varying durations ranging from 1.5 to 7 hours. The occupancy rate varies randomly between 0.25, 0.5, and 0.75 during occupied periods. No occupancy is assumed during weekends, from December 15th to January 15th, and throughout July and August. Regarding the window schedule, windows are allowed to be opened on weekdays from 9 am to 7 pm, with an open durations ranging from 30 minutes to 7 hours. Window status can change every 30 minutes. Similar to occupancy, windows are not opened during weekends, from December 15th to January 15th, and in July and August. Following the simulation, seven parameters are selected in the datasheet include electricity consumption, heat consumption, indoor temperature, occupancy schedule, outdoor temperature, solar gain, window status. With the new window open schedule, we have totally 8131 openings. In this case, data is no more highly imbalance, window opening occupied 23% of the total dataset.

4.3.2.2 Window status detection in new data

To create 2D representations of the dataset, similar to the first case study, we utilized UMAP. In Figure 4.21, the UMAP representation reveals a more balanced dataset than the one of the first case study, although there are no distinct characteristics such as distinct clusters or outlier points for the two types of data.

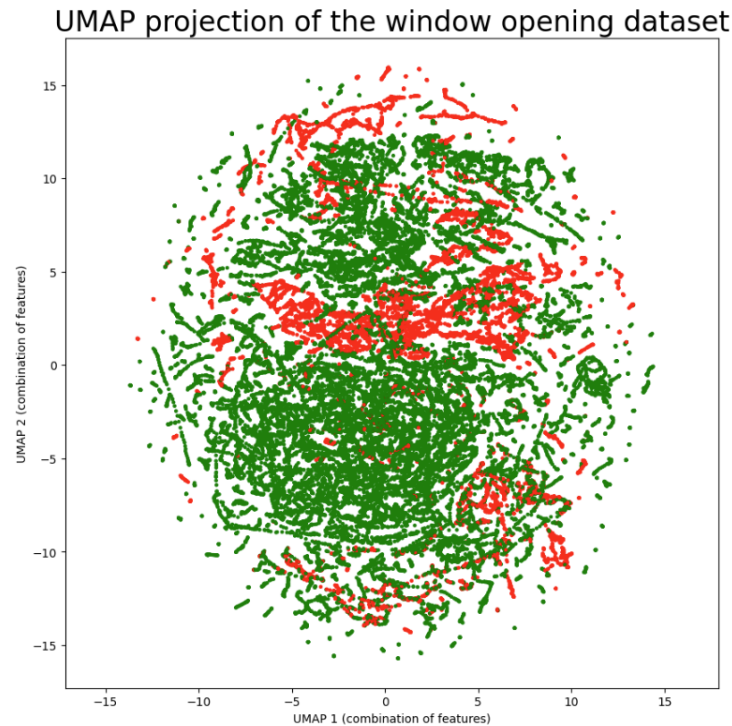


Figure 4.21 – UMAP window status for balanced case

4.3.2.3 Comparison of the unsupervised methods

Typically, balanced datasets often offer a benefit for many machine learning tasks. However, in certain cases, particularly in unsupervised learning scenarios, a balanced dataset can lead to challenges in finding underlying patterns. Imbalanced datasets often contain rarer classes or clusters that carry unique and distinctive patterns, which models can learn to identify. As datasets become more balanced, these unique patterns might become diluted or harder to discern amidst a wider array of patterns.

For instance, DBSCAN, as clustering algorithm, excels when clusters exhibit similar densities and are separated by low-density regions. However, it struggles when clusters have significantly different densities. Adjusting the epsilon (ϵ) and min_samples parameters appropriately for all clusters becomes challenging in such cases. For example, consider a case where there are two clusters, one with high density and another with low density. If epsilon (ϵ) and min_samples are set to values that are suitable for the high-density cluster, the low-density one may be completely missed, classifying it as noise. Conversely, setting these parameters for the low-density cluster could result in over-segmenting the high-density cluster into multiple smaller clusters, as the algorithm would

consider even small variations in the high-density region as separate clusters. Instead of categorizing DBSCAN as suitable for either balanced or imbalanced datasets, it is more accurate to say that its effectiveness depends on the density and separation characteristics of the clusters within the data.

As far as PCA is concerned, it exhibits limitations when dealing with data that are not linearly separable or when the variance captured by its principal components fails to align with class boundaries. Notably, PCA lacks consideration for class labels when calculating its components. On the other hand, MGD assigns each data point to Gaussian distributions with associated probabilities. MGD's strength lies in its capacity to model diverse variance-covariance structures, rendering it a versatile tool for real-world data, irrespective of whether the dataset is balanced or imbalanced.

The results of the three unsupervised ML algorithm for the new scenario are given below. In this scenario, parameters and threshold have been selected by the grid search as well to find their optimal values.

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
DBSCAN	20962	5947	115	8016
PCA	10009	16900	0	8131
MGD	18307	8602	2779	5352

Table 4.13 – Comparison of confusion-matrix-based indicators for DBSCAN, PCA, and MGD methods after Grid search

	Precision	Recall	F1 Score
DBSCAN	0.986	0.574	0.726
PCA	1	0.325	0.490
MGD	0.658	0.384	0.485

Table 4.14 – Comparison of different algorithm precision, Recall and F1

From Table 4.14, it can be observed that DBSCAN outperforms the other methods in terms of the F1 score and demonstrates a remarkably high precision, along with a moderate recall. This indicates that DBSCAN is particularly effective in correctly identifying true positives while maintaining a low rate of false positives, offering superior overall performance on this dataset compared to PCA and MGD.

Contrary to the typical trade-off between precision and recall, PCA excels in precision with a perfect score of 1.0 but shows a relatively lower recall. Its perfect precision implies

that when PCA identifies a data point as an anomaly, it is highly likely to be correct. However, its lower recall, compared to DBSCAN, indicates that PCA misses a significant number of true anomalies, despite its high accuracy in the anomalies it does detect.

MGD exhibits a moderate level of precision, recall, and F1 score. This suggests a balanced rate of errors, where it neither excels in identifying all true anomalies nor in avoiding false positives. Its performance, with a lower recall compared to DBSCAN and PCA, and a precision that is not as high as PCA's, indicates a moderate capability in distinguishing between normal data points and anomalies.

4.3.2.4 Comparison of the supervised methods

As previously shown, both the original ANN and XGBoost models demonstrated excellent performance, obviating the need for additional imbalance processing techniques. Additionally, logistic regression with oversampling or with applying SMOTE presents commendable performance.

	Precision	Recall	F1 Score
Logistic Regression (Original)	0.93	0.87	0.90
Logistic Regression (Oversampling)	0.82	0.91	0.87
Logistic Regression (SMOTE)	0.82	0.91	0.87
ANN (Original)	0.91	0.93	0.92
XGBoost (Original)	0.93	0.87	0.90

Table 4.15 – Comparison of Performance Metrics for Different Models and Sampling Techniques

From Table 4.15, we can see that logistic regression achieves a balanced accuracy score of 0.93, indicating that accuracy does not favor the majority class due to the balanced nature of the dataset. Moreover, precision, recall, and F1 score trend towards perfection for logistic regression. As far as ANN algorithm is concerned, its precision, recall, and F1 score for ANN do not perform as well as with the imbalanced dataset. The learning curve in the case of ANN shown in Figure 4.22 suggests overfitting in the model. This may be attributed to the inherent complexity of ANN, which possesses more capacity than required for the problem at hand. One potential solution is the introduction of dropout layers during training. Dropout means temporarily deactivating a random subset of neurons, along with all their incoming and outgoing connections, meaning they do not contribute to the training in that particular forward and backward pass. This regularization technique

can help mitigate overfitting and improve model generalization. Finally, the XGBoost model demonstrates strong performance on this dataset (as for imbalanced dataset), with slightly varying metrics. On the second more balanced dataset, the model has slightly lower precision but a marginally higher recall, leading to an F1 score of 0.9. In contrast, on the highly imbalanced dataset, the model achieves perfect precision but has a slightly reduced recall.

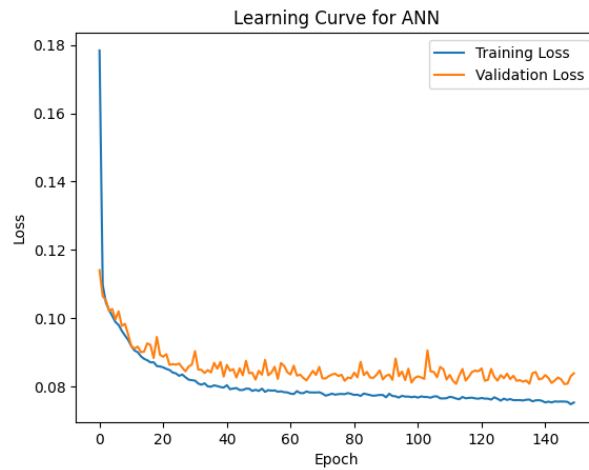


Figure 4.22 – Learning curve ANN

4.3.2.5 Semi-supervised method result

Table 4.16 and 4.17 present the results for the AutoEncoder applied to the new balanced simulation data. The model demonstrates a moderate level of precision, with a score of approximately 0.52. This suggests that while the AutoEncoder is fairly accurate in identifying true anomalies, it does encounter a notable number of false positives. However, the recall is quite high at approximately 0.97, indicating that the model is very effective in detecting the true anomalies present in the data. This could be attributed to its ability to learn and recognize patterns from the normal state, allowing it to distinguish anomalies with high accuracy. The F1 Score, at approximately 0.68, reflects a balance between precision and recall

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
AutoEncoder	5355	24	781	848

Table 4.16 – Confusion matrix for AutoEncoder in balanced simulation data

Metric	Score
Precision	0.52
Recall	0.97
F1 Score	0.68

Table 4.17 – Performance Metrics for AutoEncoder in balanced simulation data

4.3.3 Occupant presence detection

The second, "more balanced" database also simulates the influence of occupants in addition to the window open/close effect. Therefore, this database can be utilized to further explore the effectiveness of the previously employed methods in detecting occupancy. The analysis, centered on the same aspects, will be expedited here since the trends observed previously are largely confirmed.

4.3.3.1 Comparison of the unsupervised methods

Table 4.18 and Table 4.19 show the comparison result of the unsupervised methods (DBSCAN, PCA, and MGD). Parameters and threshold for the three methods have been updated by the grid search in here as well.

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
DBSCAN	11446	12929	3627	7038
PCA	1	24374	0	10665
MGD	16209	8166	4825	5840

Table 4.18 – Comparison of confusion matrix results for DBSCAN, PCA, and MGD methods after Grid search

	Precision	Recall	F1 Score
DBSCAN	0.660	0.352	0.460
PCA	0.304	1	0.467
MGD	0.548	0.417	0.473

Table 4.19 – Comparison of different algorithm precision, Recall and F1

Based on the results obtained with the unsupervised methods presented in Table 4.18 and Table 4.19, it is evident that among the three methods, PCA achieves the

highest precision with a perfect score of 1.0, but this comes with a relatively lower recall. DBSCAN, while not achieving the perfect precision of PCA, strikes a balance between precision and recall. MGD, although it does not reach the precision level of PCA, performs moderately in terms of both precision and recall.

4.3.3.2 Comparison of the supervised methods

The performance metrics presented in Table 4.20 for various models and sampling techniques on the second simulation dataset offer several noteworthy insights.

The XGBoost model stands out, exhibiting superior performance in terms of precision, recall, and F1 score among all the models evaluated. This underscores XGBoost's efficacy for this specific dataset, both in terms of accurately identifying positive instances and in capturing a significant proportion of actual positive instances. The robustness of XGBoost can be attributed to its gradient boosting mechanism, which iteratively refines the model's predictions by focusing on instances that were previously misclassified. This iterative refinement, combined with its capability to handle missing data and its inherent resistance to overfitting, makes XGBoost a compelling choice for this dataset.

The ANN (Original) model also demonstrates commendable performance, particularly in terms of recall, emphasizing its capability in identifying the majority of actual positive cases. However, the computational complexity and the potential for overfitting in deep neural networks might make XGBoost a more favorable choice in scenarios where interpretability, scalability, and computational efficiency are paramount.

Logistic Regression, when applied to the original dataset, achieves a harmonious balance between precision and recall, culminating in a respectable F1 score. Interestingly, its performance experiences a slight enhancement in terms of the F1 score when oversampling and SMOTE techniques are employed. However, this comes at the cost of a minor dip in precision. For the Logistic Regression model, the performance metrics remain invariant whether the Oversampling or SMOTE technique is applied. This observation underscores the equivalent efficacy of both techniques in addressing class imbalance for this dataset-model combination.

In conclusion, while all models present specific strengths, the XGBoost model's combination of accuracy, efficiency, and scalability makes it a particularly compelling choice for this dataset. But the selection of the model and the sampling technique can profoundly influence the performance metrics, and it is imperative to calibrate the model choice based on the specific exigencies of the application.

	Precision	Recall	F1 Score
Logistic Regression (Original)	0.74	0.64	0.69
Logistic Regression (Oversampling)	0.66	0.76	0.71
Logistic Regression (SMOTE)	0.66	0.76	0.71
ANN (Original)	0.79	0.83	0.81
XGBoost	0.84	0.81	0.825

Table 4.20 – Comparison of Performance Metrics for Different Models and Sampling Techniques

4.3.3.3 Semi-supervised method results

Based on the provided confusion matrix and performance metrics, presented in Tables 4.21 and 4.22, respectively, the AutoEncoder model shows moderate performance in detecting presence. Its precision score of approximately 0.559 suggests that while the model is reasonably accurate in classifying a case as 'presence', it does encounter a number of false positives. However, the recall score of approximately 0.677 indicates that the model is quite effective in capturing actual 'presence' cases, missing fewer instances than implied by the term 'substantial'. Overall, the model strikes a balance between precision and recall, as reflected in the F1 score of approximately 0.612.

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
AutoEncoder	4284	573	949	1202

Table 4.21 – Confusion matrix for the AutoEncoder for presence detection

Metric	Score
Precision	0.559
Recall	0.677
F1 Score	0.612

Table 4.22 – Performance Metrics for the AutoEncoder for presence detection

4.3.4 An intermediate conclusion following case studies with simulation data

In the previous sub-sections (4.3.1 to 4.3.3), we conducted an extensive comparative analysis involving a combination of unsupervised, supervised, and semi-supervised algorithms applied on two distinct simulation datasets. One dataset exhibited a high level of

class imbalance and an unrealistic scenario, primarily designed for testing purposes, while the second represented a more balanced and realistic scenario. For the unsupervised methods, for both simulation cases, we employed three learning methods: DBSCAN, PCA, and MGD. Among them, DBSCAN consistently demonstrated the highest performance metrics, including precision, recall, and F1 Score, on both datasets. This notable performance can be attributed to DBSCAN's proficiency in effectively identifying clusters of varying shapes and sizes, making it particularly advantageous for handling imbalanced datasets. However, it is important to note that this superior performance came at the expense of computational speed, as DBSCAN was the slowest among the tested algorithms. This limitation renders it less suitable for real-time applications or the analysis of extensive datasets.

On the contrary, PCA proved to be computationally efficient but its linear nature and emphasis on capturing variance make it less suitable for classifying scenarios with intricate or poorly separated class boundaries. It consistently produced a high number of false negatives in exchange for good recall. This behavior could be attributed to PCA's approach of capturing variance in the dataset. Principal components derived from PCA may not align well with class boundaries, potentially blurring distinctions between different classes. In the specific context of window opening detection, if PCA's principal components predominantly capture factors other than the open/closed state of windows, the algorithm may struggle to classify this feature accurately, resulting in a high rate of false negatives.

MGD consistently achieved high precision but exhibited lower recall. This characteristic can be linked to the foundational principles and assumptions of MGD. MGD operates on the assumption that class features follow a Gaussian distribution. Its high precision indicates MGD's effectiveness in correctly identifying 'open' window cases as such, resulting in a lower rate of false positives. However, the lower recall suggests that MGD might be missing a significant number of actual 'open' cases, leading to a higher number of false negatives. The underlying reasons for this behavior are rooted in MGD's Gaussian distribution assumptions. If the Gaussian distributions for 'open' and 'closed' states do not have a clear separation, MGD might struggle to detect all 'open' cases, which results in lower recall. The degree of conformance of the dataset to the Gaussian distribution assumptions and the distinctness between 'open' and 'closed' categories in the feature space significantly impact MGD's ability to discern between these two states effectively.

To conclude, we have chosen DBSCAN as the potential algorithm, although its com-

putational requirements are a drawback, and we will explore acceleration methods in a later section of this chapter.

For the supervised learning algorithms, for both simulation cases, we used XGBoost, ANN, and Logistic Regression. Logistic Regression initially underperformed on the imbalanced dataset but demonstrated significant improvement with the application of techniques such as oversampling and SMOTE. Despite achieving near-perfect performance metrics post-resampling, it faced challenges in a more complex simulation case study. These challenges could be attributed to its inherent limitations in capturing non-linear relationships and sensitivity to class imbalance.

Conversely, XGBoost showcased robust performance across both datasets. Although it wasn't flawless on the imbalanced dataset, its effectiveness significantly improved when complemented with resampling techniques like oversampling and SMOTE. XGBoost's versatility enables it to handle balanced and imbalanced datasets efficiently, accommodating both linear and non-linear patterns. While it may not be as fast as Logistic Regression, it remains well-suited for medium-to-large datasets where some level of model complexity is acceptable.

Finally, ANN emerged as the most accurate but computationally intensive algorithm. It excelled in capturing complex, non-linear relationships, making it the most accurate model for both datasets. However, its high accuracy came at the cost of computational speed, rendering it less suitable for real-time or resource-constrained scenarios.

Thus, for the supervised methods, XGBoost emerges as the potentially most appropriate algorithm, while ANN may be suitable for simulations without stringent time constraints.

Regarding semi-supervised methods, their performance was not as impressive as the supervised methods and did not significantly outperform the unsupervised methods. Additionally, considering the multi-activities detection requirements, we will not consider semi-supervised methods as potential algorithms for our case study.

We finally state that DBSCAN and XGBoost emerge as the top choices for each respective category. Further discussions on acceleration methods for these two algorithms will be covered in a later section.

4.4 Real case study results

4.4.1 Presentation of the real case dataset

In this section, we will discuss a real-world case study using data collected from a computer lecture room at Polytech Angers. Within this lecture room, fourteen combined ambient sensors were installed to determine the optimal sensor location. As we identified the optimal sensor (sensor 101) in the previous chapter, our focus in this study is solely on utilizing this optimal sensor's data. After collecting and preprocessing the data, we obtained over eight parameters and 80,000 data points for the period between October 1, 2022, and December 1, 2022. Among the eight parameters, we carefully selected those with a strong correlation to window status, including ambient data from sensor 101 (CO₂, TVOC, Humidity, Light, Sound, Temperature), outside temperature, and window status itself. It is worth noting that some data points were missing. To address these gaps in the data, we employed a strategy of imputing missing values by taking the mean of data points before and after the missing values. If two or more consecutive lines of data are missing, the entire row is removed from consideration. After processing, we were left with 17,000 data points at 5-minute intervals for the specified period.

To gain insights into the data distribution corresponding to window status, we employed UMAP for dimensionality reduction, resulting in Figure 4.23. The plot shows two intricately intertwined datasets, indicating the complex relationships and interdependencies between them. The balance between the two types of data in the plot suggests that they share a high degree of similarity in the dimensional space visualized. The observed overlap in the UMAP plot serves as an indication that the selected features do not easily differentiate between the two types of data. This observation might inform us about the underlying structure of the data and lead to further investigations.

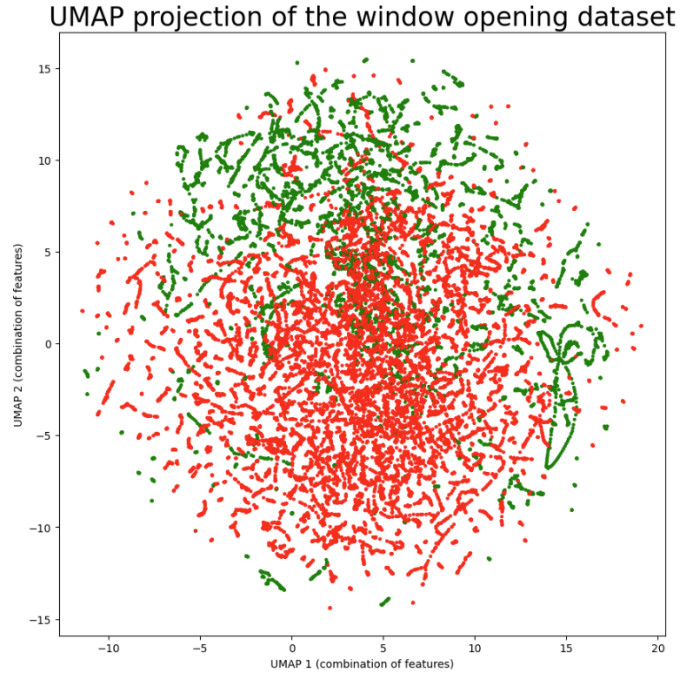


Figure 4.23 – UMAP in real data

4.4.2 Comparison of learning algorithms - Main results for real case study

4.4.2.1 Comparison of the unsupervised methods

The results of the unsupervised method are presented in Table 4.23 and Table 4.24. For each method, hyperparameters or thresholds have been determined through grid search, as used previously. In the results, PCA and MGD exhibit extremely low True Negative (TN), indicating that these models tend to classify almost everything as the positive class.

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
DBSCAN	401	6498	417	14860
PCA	0	6899	1	15276
MGD	1	6898	0	15277

Table 4.23 – Comparison of confusion matrix results for DBSCAN, PCA, and MGD methods after Grid search

	Precision	Recall	F1 Score
DBSCAN	0.973	0.696	0.811
PCA	1	0.689	0.816
MGD	1	0.689	0.816

Table 4.24 – Comparison of different algorithm precision, recall and F1 in real data

One primary reason for this behavior is the class imbalance present in the dataset. Specifically, there are 15,277 instances of windows being open ('1') and only 6,899 instances of them being closed ('0'). Consequently, many instances are classified as open, resulting in high recall but low precision, especially for the minority class (closed windows). However, this imbalance alone cannot explain the entire issue, as the simulation dataset is also highly imbalanced. To further understand the challenge posed by the data, we can examine the feature distributions shown in Figures 4.24 to 4.27. In these Figures, the y-axis represents the density of data points, with higher density meaning more data points falling within that range. The x-axis depicts the value range for each feature. These figures display the characteristics of the first four features (CO₂, TVOC, temperature and humidity), categorized by window state (open or closed). Notably, the feature distributions exhibit significant overlap between the two classes ('Open' and 'Closed'). This overlap could make it difficult for algorithms to effectively distinguish between the two states.

In summary, despite DBSCAN gives relatively high false negatives, it still outperforms other unsupervised methods in this challenging environment. This superiority can be attributed to DBSCAN's strengths, such as its ability to handle noise and outliers, its lack of a predefined number of clusters, and its neighborhood-based search approach.

4.4.2.2 Comparison of the supervised method

Table 4.25 presents the performance of supervised methods. Among the supervised methods, both ANN and XGBoost still emerge as the most suitable algorithms for this dataset, as they consistently deliver high precision, recall, and F1 scores. Furthermore, when we take into account computational complexity, XGBoost remains the preferred choice in this context. While Logistic Regression demonstrates relative effectiveness, it falls short in comparison to the performance achieved by ANN and XGBoost. Additionally, the application of oversampling and SMOTE techniques does not appear to yield significant

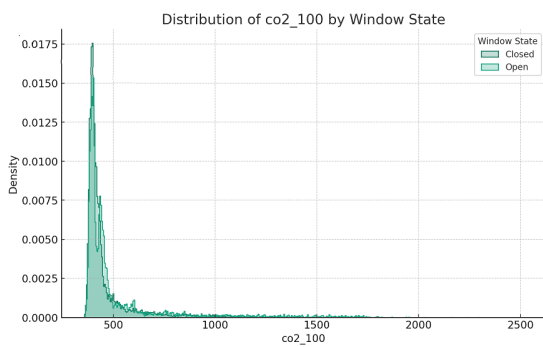


Figure 4.24 – Features distribution

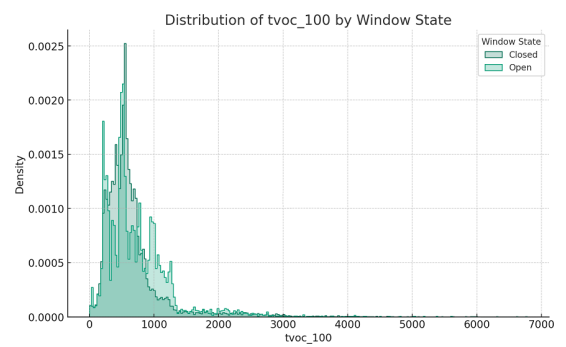


Figure 4.25 – Features distribution

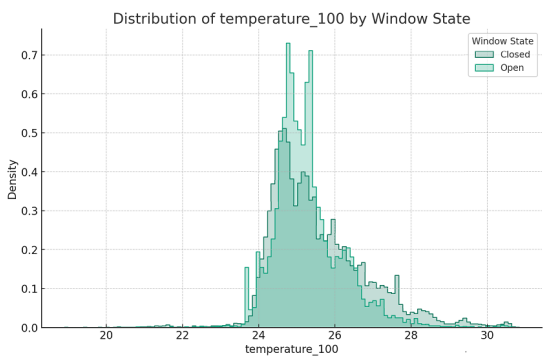


Figure 4.26 – Features distribution

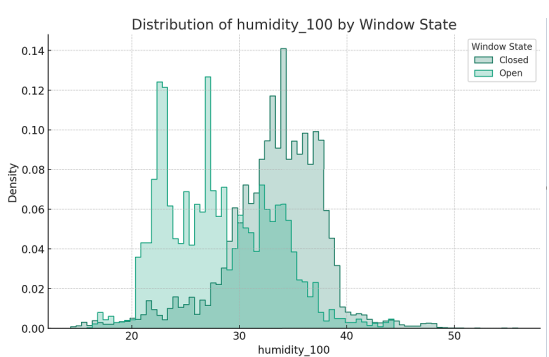


Figure 4.27 – Features distribution

benefits over the original logistic regression model for this particular dataset.

	Precision	Recall	F1 Score
Logistic regression (Original)	0.82	0.90	0.87
ANN (Original)	0.95	0.97	0.96
Logistic Regression (Oversampling)	0.88	0.79	0.83
Logistic Regression (SMOTE)	0.88	0.79	0.83
XGBoost (Original)	0.99	0.99	0.99

Table 4.25 – Comparison of Performance Metrics for Different Models and Sampling Techniques

4.4.2.3 Semi-supervised method results

In Tables 4.26 and 4.27, We observe that the AutoEncoder demonstrates an impressively high recall score of 0.993, indicating that it successfully identifies nearly all actual 'positive' instances. This high recall suggests that when a 'positive' instance occurs, the model is almost always able to detect it. However, the model's precision score is extremely low at 0.044, meaning it generates a significant number of false positives. In essence, while the model is adept at identifying positives, it also incorrectly labels many instances as positive. The F1 score, the harmonic mean of precision and recall, is consequently low at 0.085, highlighting the model's lack of balance and overall poor performance. Despite its ability to detect most positive instances, the AutoEncoder's excessive rate of false positives makes it impractical for effective use in this context.

Method	True Negative (TN)	False Negative (FN)	False Positive (FP)	True Positive (TP)
AutoEncoder	1399	1	2901	135

Table 4.26 – Confusion matrix for the AutoEncoder for real case

Metric	Score
Precision	0.044
Recall	0.993
F1 Score	0.085

Table 4.27 – Performance metrics for the AutoEncoder for real case

4.4.2.4 Ending note

For the real case study, DBSCAN and XGBoost remain our selected algorithms. In this positively imbalanced dataset, both DBSCAN and XGBoost have exhibited strong performance in both unsupervised and supervised methods. Striking a balance between accuracy and computational complexity, we have ultimately chosen DBSCAN and XGBoost algorithms.

4.5 Algorithms acceleration

Incorporating occupant behavior detection as a feedback signal in a building control system necessitates a swift detection process, particularly in real-time scenarios. Occupant comfort can be highly dynamic, and rapid adjustments based on real-time occupancy data can minimize energy usage, leading to improved user experience and satisfaction. Therefore, to be optimized for real-time usage, the previously selected algorithms can or should be "accelerated"

4.5.1 DBSCAN acceleration

DBSCAN poses computational challenges because it requires comparing each data point with every other point to identify neighbors. It exhibits a computational complexity of $O(n^2)$, where n represent number of observation. Even with an indexing structure such as a k-d tree is used to speed up the region queries, it retains a computational complexity of $O(n \log n)$. As seen in the previous section, DBSCAN was shown as the slowest among the unsupervised methods.

Without compromising detection accuracy, a precomputed distance matrix approach is used associated with DBSCAN algorithm. Typically, DBSCAN calculates the distance between each pair of data points, which can be time-consuming. So, the precomputed distance matrix is a matrix where the distance between each pair of points is calculated in advance and stored. When the DBSCAN requires distance calculations, it can use this precomputed matrix, avoiding redundant distance calculations and significantly speeding up the process. With this method, the calculation time for DBSCAN decreases from 6 seconds to a mere 0.0035 seconds¹⁵.

15. All the time computations are obtained using the environment of Intel Core i7-3630 CPU @2.50GHz, 8G RAM.

4.5.1.1 DBSCAN parameter selection acceleration

In addition to DBSCAN acceleration techniques, optimizing the selection of parameters such as *eps* and *minPts* to enhance clustering quality is a critical focus for future research, as noted by Wen et al. (2021). However, conducting parameter optimization via Grid search is computationally expensive and time-consuming. Thus, we have developed an optimized methodology based on our grid search approach for parameter selection in DBSCAN that significantly reduces computation time without compromising clustering quality.

The methodology comprises two main steps: a "coarse-to-fine" parameter search and data pre-filtering of data.

Step 1: Coarse-to-Fine Parameter Search Rather than searching the entire parameters space, we use a two-stage grid search strategy. The coarse-to-fine approach enables us to efficiently focus on a smaller, more pertinent area within the parameter space. And by conducting a fine search around the best results from the coarse search, the method ensures a good trade-off between computational effort and quality of clustering.

Coarse Search:

Coarse Search: Initially, the algorithm explores a broad range of parameter values to find a "good enough" solution quickly. This phase gives us an initial idea of what parameter values may be effective without having to search the entire parameter space exhaustively.

1. **Select Ranges:** Define broad ranges for ϵ and `min_samples`.
 - $\epsilon = [0, 0.3, \dots, 0.9]$
 - `min_samples` = [5, 10, ..., 15]
2. **Grid Search:** Perform a standard grid search on this coarse grid to identify the best parameters.

$$\text{Best F1 score} = \max_{\epsilon, \text{min_samples}} F1(\epsilon, \text{min_samples}) \quad (4.19)$$

Fine Search:

Based on the results of the coarse search, a narrower range is defined around the most promising parameter values. A more detailed search is conducted within this refined range.

1. **Narrow Down Ranges:** Based on the best parameters (ϵ^* , `min_samples*`) from the coarse search, define a finer grid around these values.
 - $\epsilon = [\epsilon^* - 0.1, \epsilon^*, \epsilon^* + 0.1]$

— `min_samples = [min_samples* - 1, min_samples*, min_samples* + 1]`

2. **Grid Search:** Perform grid search on this fine grid.

$$\text{Best F1 score} = \max_{\epsilon, \text{min_samples}} F1(\epsilon, \text{min_samples}) \quad (4.20)$$

Step 2: Pre-filtering (Optional) Pre-filtering techniques can be applied to the dataset to remove outliers or irrelevant points before performing the DBSCAN clustering. This can further speed up the process. Quantile Filtering can be employed as a pre-processing step to remove potential outliers in the dataset. Let X be our dataset with n features. The pre-filtering can be formulated mathematically as follows:

For each feature f_i , $i = 1, \dots, n$, do the following:

1. Calculate the 1st quartile ($Q1_i$) and 3rd quartile ($Q3_i$):

$$Q1_i = \text{Quantile}(X_{f_i}, 0.25), \quad Q3_i = \text{Quantile}(X_{f_i}, 0.75) \quad (4.21)$$

2. Compute the Interquartile Range (IQR_i):

$$IQR_i = Q3_i - Q1_i \quad (4.22)$$

3. Determine the Lower Bound (LB_i) and Upper Bound (UB_i):

$$LB_i = Q1_i - 1.5 \times IQR_i, \quad UB_i = Q3_i + 1.5 \times IQR_i \quad (4.23)$$

4. For each data point x in X , remove x if any of its features f_i fall outside the bounds:

$$x = \{x \in X \mid \forall i, LB_i \leq x_{f_i} \leq UB_i\} \quad (4.24)$$

By applying this quantile filtering, we can effectively remove outliers based on the IQR method for each feature, thus preparing the data for more accurate clustering by the DBSCAN algorithm.

4.5.1.2 Result:

The optimized methodology showed significant improvements in computation time while maintaining the quality of clustering. The performance are as follows:

Metric	Standard Grid Search	Optimized Search
Time (seconds)	415.64	145.78 (65% reduction)
Best F1 score	0.4595	0.4595
Best ϵ	0.1	0.1
Best <code>min_samples</code>	14	14

Table 4.28 – Comparison of Standard and Optimized Search on Simulation Dataset

From the results, it is evident that the optimized search process can significantly reduce computation time by 65%, all without compromising the quality of clustering, as evidenced by the maintained F1 score. This method proves particularly advantageous for large datasets where a conventional grid search would be prohibitively expensive. Therefore, it serves as a valuable tool for applications demanding efficient and effective clustering processes.

4.5.2 XGBoost acceleration

In gradient boosting, multiple trees are constructed sequentially, each correcting the errors of its predecessor, ultimately improving predictive performance. However, constructing optimal decision trees is a computationally challenging task.

4.5.2.1 XGBoost parameter selection acceleration

Our original algorithm employs exact tree learning, a traditional method for growing decision trees in a greedy manner. This approach involves iterating through all features and their possible splits to find the best split based on a predefined objective function. The objective function combines a loss function and a regularization term. The objective function aims to minimize:

$$\text{Obj} = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{j=1}^{|T|} \Omega(f_j) \quad (4.25)$$

Here $\Omega(f_j)$ typically is a function of the structure of the tree T and the output score at leaf j . It is often written as:

$$\Omega(f) = \gamma|T| + \frac{1}{2}\lambda \sum_{j=1}^{|T|} w_j^2 \quad (4.26)$$

To find the optimal structure, the algorithm explores every possible split for each feature, leading to exhaustive computation.

Starting from a root node containing all data points, Exact Tree Learning iteratively considers each feature and evaluates all potential split points. The algorithm quantifies the efficacy of each split using metrics such as "gain" or "information gain." The gain from a split point s on feature j is calculated as:

$$\text{Gain}(s) = \frac{1}{H_L + \lambda} \left(\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right) \quad (4.27)$$

After identifying the most effective split, the data is partitioned into two subsets accordingly. This process of evaluation and splitting is recursively applied to each new child node.

While this method is more likely to yield a good result, it is computationally intensive, especially for datasets with numerous features or high cardinality.

An alternative to exact tree learning is histogram-based tree learning. This approach prioritizes computational efficiency by approximating the continuous feature space. Each feature is discretized into discrete bins:

$$\text{Bin}(x_j) = \operatorname{argmin}_{b \in k} \|x_j - b\| \quad (4.28)$$

Histograms are created based on these bins, reducing the number of potential split points. The algorithm then looks for the best split based on aggregated gradients and Hessians of the bins, forming G_{bin} and H_{bin} .

Just like in Exact Tree Learning, the objective remains the same: to minimize the objective function. However, the algorithm operates on these pre-aggregated bins, making it computationally less demanding. The tree starts at the root node and evaluates the aggregated gradients and Hessians for each bin. The best split is chosen based on these aggregated values, and the data is divided into two child nodes. This process is then recursively applied to each of the new child nodes. In this method, the same Gain function as in the exact method is used, but the gain is calculated for each bin instead of each data point:

$$\text{Gain}(\text{Bin}) = \frac{1}{H_{\text{bin},L} + \lambda} \left(\frac{G_{\text{bin},L}^2}{H_{\text{bin},L} + \lambda} + \frac{G_{\text{bin},R}^2}{H_{\text{bin},R} + \lambda} - \frac{(G_{\text{bin},L} + G_{\text{bin},R})^2}{H_{\text{bin},L} + H_{\text{bin},R} + \lambda} \right) \quad (4.29)$$

While this method sacrifices some accuracy due to approximations, it significantly reduces computational demands.

Another solution is the XGBoost-Ray, which is an innovative backend designed to support distributed learning in XGBoost. It takes advantage of Ray, a powerful framework tailored for distributed computing. XGBoost-Ray facilitates XGBoost training across a cluster of hundreds of nodes, offering scalability, fault resilience, flexible training capabilities, and seamless integration with the Ray Tune library for hyperparameters optimization.

Unlike the standard XGBoost library, which is limited to a single CPU or GPU on one machine, XGBoost-Ray extends these limitations by enabling the use of multiple processors and GPUs across multiple machines. This becomes essential for handling extensive datasets that cannot fit into the memory of a single computing unit, making distributed training indispensable.

The architecture of XGBoost-Ray enables distributed learning to occur across a multi-node, multi-GPU cluster, enhancing the capability and speed of the training process.

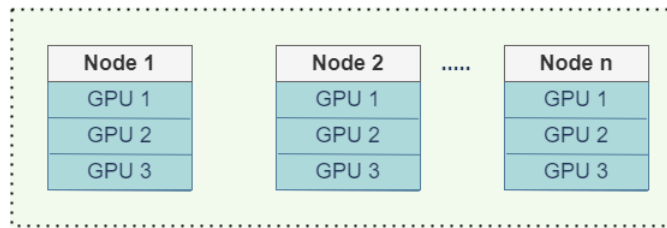


Figure 4.28 – Architecture of XGBoost-Ray

During the training phase, XGBoost-Ray sets up a group of training actors that cover the entire computational cluster. Each of these actors takes charge of training the model on a specific segment of the dataset. This methodology is referred to as data-parallel training. To ensure that all actors contribute to improving the model collectively, they exchange gradient information using a tree-based AllReduce algorithm, as illustrated in the subsequent Figure 4.29.

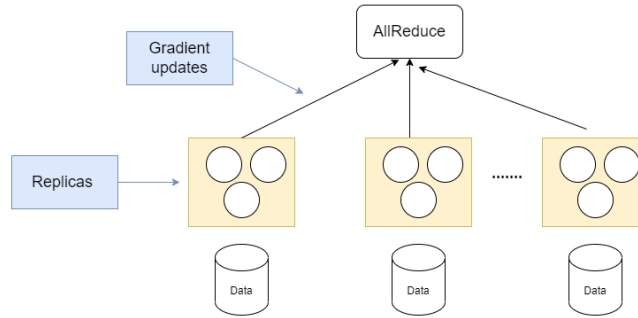


Figure 4.29 – XGBoost-Ray training actors

4.5.2.2 Result for XGBoost acceleration

The XGBoost acceleration method with different dataset will be summed up in the Table 4.29 below. The data reveals that the histogram tree learning method offers a remarkable 85% improvement in computation time and efficiency without significantly compromising performance. In contrast, the XGBoost-ray method did not consistently achieve the expected results. While XGBoost-ray should theoretically be faster, especially when making efficient use of parallelization and GPU acceleration, certain factors can impede its performance. Notably, when dealing with smaller datasets, data transfer to the GPU can counteract the speedup gained through parallelization. Additionally, overheads related to initializing and communicating with Ray and GPU acceleration can extend runtimes.

Dataset	Method	Training Time (s)	Precision	Recall	F1-Score
Imbalanced Simulation	Distributed	5.78	1.0	0.5833	0.7368
	Origin	1.33	1.0	0.8966	0.9455
	Hist	0.18	1.0	0.7083	0.8293
Real	Distributed	4.34	0.9759	0.9733	0.9746
	Origin	3.60	0.9951	0.9944	0.9947
	Hist	0.29	0.9960	0.9941	0.9951
Balanced Simulation	Distributed	8.01	0.9580	0.7974	0.8704
	Origin	2.19	0.9266	0.8674	0.8960
	Hist	0.28	0.9361	0.8631	0.8981

Table 4.29 – Performance metrics for different training methods across various datasets.

4.6 Conclusion of the chapter

This fourth chapter provides an extensive comparison of various machine learning and statistical methods. It outlines the selection process for candidate algorithms based on their respective strengths and weaknesses. Specifically, we have chosen DBSCAN, PCA, and MGD for unsupervised methods; Logistic Regression, ANN, and XGBoost for supervised methods; and AutoEncoder for semi-supervised methods. These algorithms were selected for their suitability in predictive modeling across different datasets, including imbalanced simulations, balanced simulations, and real-world data, with a primary focus on detecting specific building activities, notably window status. Each algorithm was applied meticulously, taking into account their underlying assumptions and the need for appropriate parameter settings. For the unsupervised methods, DBSCAN demonstrated promising results across all three datasets, especially excelling in the case of imbalanced data. Conversely, PCA and MGD consistently produced high false negatives due to their tendency to classify a significant portion of data as positive. DBSCAN's strengths, such as its ability to discover clusters of arbitrary shapes without requiring the user to predefine the number of clusters, were noteworthy. However, it faces challenges when dealing with clusters of varying densities. In the supervised methods context, the ANN exhibited the capacity to learn complex, non-linear relationships, delivering strong performance. Nonetheless, it demands substantial computational resources and time, particularly for larger networks, and is susceptible to overfitting. Logistic Regression, on the other hand, is quick to implement and train but is less effective in handling complex, non-linear relationships. XGBoost, by combining the strengths of both, proves capable of managing complex systems efficiently without excessive computational costs. Within the semi-supervised approach, the AutoEncoder stands out for its proficiency in learning representations and dimension reduction, especially when trained with a single data type.

As a result of performance evaluations and specific requirements, we identified the need for both unsupervised and supervised methods. Among unsupervised methods, DBSCAN was selected for further use in Chapter 5, in conjunction with root cause analysis, for detecting household behavior. In the supervised approach, we opted for XGBoost, which will be employed to distinguish four occupants' scenarios in Chapter 5 using a limited amount of labeled data and self-supervised learning. Throughout this chapter, we also implemented strategies to enhance computational efficiency, such as using precomputed matrices to accelerate DBSCAN by ten fold and employing histogram tree learning to

boost XGBoost's speed by nine fold.

One lesson of this fourth chapter is the importance of understanding the characteristics of the dataset, methods assumptions, and the balance between model complexity and overfitting. It emphasizes that there is no universal approach; the choice of method should align with the specific requirements and constraints of the problem at hand.

MULTI ACTIVITIES DETECTION BASED ON ROOT CAUSE ANALYSIS

Understanding occupancy behaviors is crucial for energy-efficient building operations. As discussed previously, while traditional supervised methods have achieved significant success in detecting occupancy-driven behaviors, they have some drawbacks: they rely heavily on labeled data, which can be costly, time-consuming, and intrusive. In addition, methods requiring less labeled data perform well in detecting occupation, but struggle with classifying different activities. In this chapter, firstly, we present a novel approach to classify between window-opening and occupancy presence events using only ambient sensor and thereby reducing intrusiveness. It consists in applying self-supervision for data augmentation, followed by transfer learning from the models trained on an initial, or so-called "pretext" task. The result of self-supervision has an accuracy of 86.85% to classify the behaviors. Secondly, we present another novel hybrid approach based on Hotspot algorithm to identify the root cause of the occupancy-driven behaviors. It is complemented by applying a BN to classify window-opening and occupancy presence events. The result shows a good ability to detect the multi activities under unsupervised method.

5.1 Introduction

Accurate prediction and understanding of occupants' behaviors can assist in the development of energy-efficient building management systems, such as predefined behavior schedules for control systems or real-time detection for adjusting control systems in real time. Additionally, it can help alert building users or managers to unintentionally left open windows, thereby reducing unnecessary ventilation heat losses.

Besides the privacy issue (Cody et al. 2021), methods for activity detection have been proven to be not entirely accurate. This inaccuracy can be attributed (i) to the sensor layout, (ii) to the availability of labelled data and (iii) to ability of methods to distinguish

between several activities. This first aspect, sensor layout, has been already handled in Chapter 3. Traditional supervised machine learning approaches have been successfully employed for classification tasks. However, these methods heavily rely on the availability of labeled data, which can be intrusive to obtain and often require significant manual effort. This aspect was discussed in Chapter 4 where unsupervised methods have been explored and compared. In this previous chapter, it was also shown that the non-intrusive, unsupervised, or semi-supervised methods (requiring less labeled data and manual effort) can detect the occurrence of a behavior. However, classifying and distinguishing between each behavior is more challenging, as illustrated in the PCA example in Chapter 4 (different behaviors mixed together and unable to be classified). Different activities (e.g., windows opening, people entering and leaving a room, etc.) can cause fluctuations in multidimensional attributes (e.g., temperature, CO₂, etc.). Conversely, different combinations of attributes can reflect different combinations of activities. There is generally no unique solution and no unique relationship between changes in activities and changes in attributes, making the detection process more complex.

In the present Chapter 5, our objective is then to improve multi-activities detection in buildings by addressing the related challenges: (i) intrusive sensors, (ii) labeled data, and (iii) classification problems in unsupervised methods. To tackle these challenges, we initially rely solely on ambient sensors to gather environmental information, thereby sidestepping potential privacy issues. Next, we examine the feasibility of acquiring semantic interpretations without supervision, thus eliminating the need for manual labeling of sensor data with explicit categories, such as activity types. Specifically, we aim to derive features of comparable quality to those obtained using fully supervised techniques.

One possible solution is a rising approach to feature learning called self-supervised learning. This method is presented in section 5.3 and sub-section 5.3.2, and then applied in section 5.3.2.2. Self-supervised learning method introduces auxiliary tasks, often referred to as pretext or surrogate tasks, where labels are directly extractable from the data without human intervention. By exploiting these powerful guidance cues from the surrogate tasks, we can employ objective functions similar to those used in traditional supervised learning scenarios. The field of vision has already introduced numerous self-supervised tasks to enhance representation learning from static images, videos, and sound. In the building context application, this method remains largely unexplored in the existing literature. In the present chapter, our primary contribution is the pioneering application of self-supervised learning to ambient sensory data for distinguishing different occupant

behaviors.

Another objective is to identify the underlying causes that lead to different occupant behaviors. Once we understand these root causes, we can use them to infer the likelihood of a particular behavior occurring. This represents a thorough form of unsupervised learning. In this chapter, another key contribution is the adaptation of the Hotspot algorithm to the building context for continuous time-series data. Traditionally, the Hotspot algorithm relied on predicted KPIs (Key Performance Indicators) to detect activities or faults, but its accuracy was highly dependent on a strong correlation assumption between the KPIs and root causes. To improve activity and fault detection capabilities and reduce the search space, we integrated DBSCAN with the Hotspot algorithm. Finally, to tackle the performance evaluation challenge and establish a link between root causes and activities, we implemented BNs to infer the probabilities of various activities. We also tested different network structure estimation and parameter estimation methods to achieve better accuracy. The background and methodology for this second approach is presented in sections 5.4, and sub-sections 5.4.2 and 5.4.3.

5.2 Case study

In this section, we will remind the specificity of the case studies (introduced in Chapter 2) for the multi-activity detection.

5.2.1 Simulation case study

The balanced dataset introduced in Chapter 2 is used. As a reminder, several assumptions have been made for the occupancy and window schedules. For occupancy, the room may be occupied on weekdays from 9 am to 7 pm. The duration of occupancy is random, ranging between 1.5 to 7 hours. The occupancy number varies randomly between 0.25, 0.5, and 0.75 during the occupied periods. There is no occupancy during weekends, from December 15th to January 15th, and throughout July and August. As for the window schedule, the window status can change every 30 minutes and windows can only be opened during presence time slots. After the simulation, seven parameters are selected from the datasheet, including electricity consumption, heat consumption, indoor temperature, occupancy schedule, outdoor temperature, solar gain, and window status. The plot of these parameters can be seen from Figure 5.1 to Figure 5.7. From the Figure 5.3 (indoor

temperature), we can notice that the temperature fluctuates around the heating setpoint at 22°C during the heating period. We can also observe a set back point at 20°C during the night. Additionally, we also notice that the cooling setpoint at summer is 28 °C. The simulation also indicates breaks during July and August from the window status (Figure 5.7) and occupancy status (Figure 5.4). In the latter Figure 5.4, the y-axis values indicate the heat gain from the occupants. All the values will be converted to 0 and 1 to represent the presence or absence of occupants.

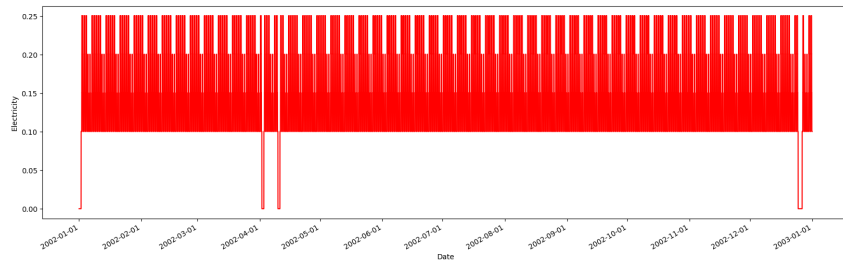


Figure 5.1 – Electricity simulation case

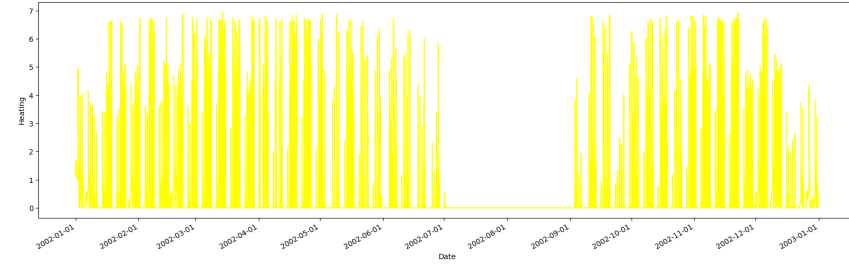


Figure 5.2 – Heat consumption simulation case

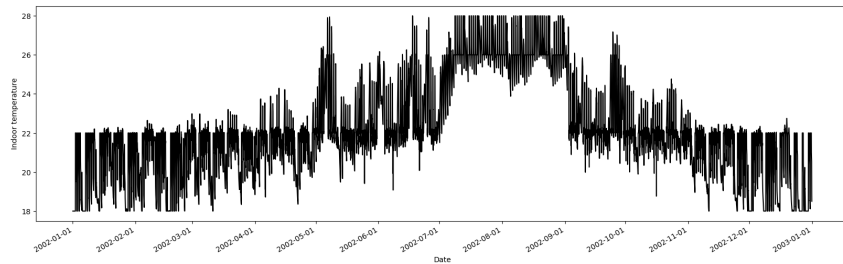


Figure 5.3 – Indoor temperature simulation case

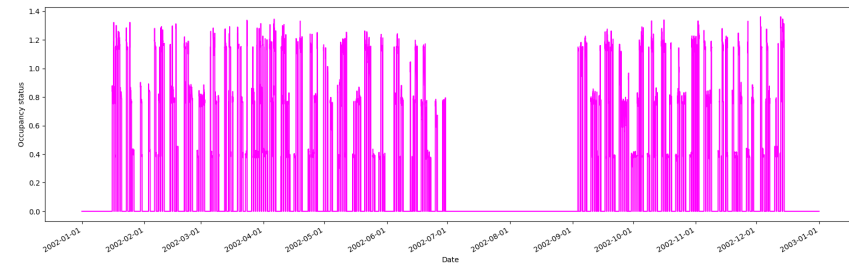


Figure 5.4 – Occupancy status simulation case

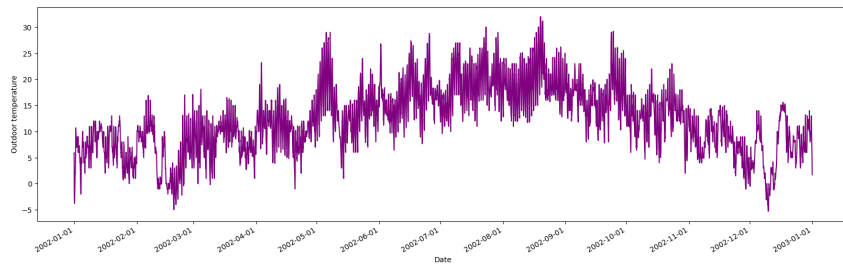


Figure 5.5 – Outdoor temperature simulation case

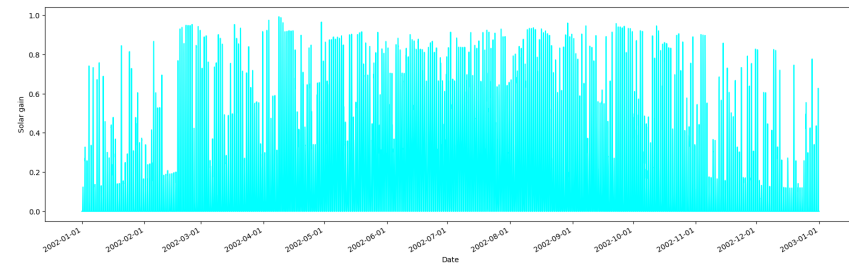


Figure 5.6 – Solar gain simulation case

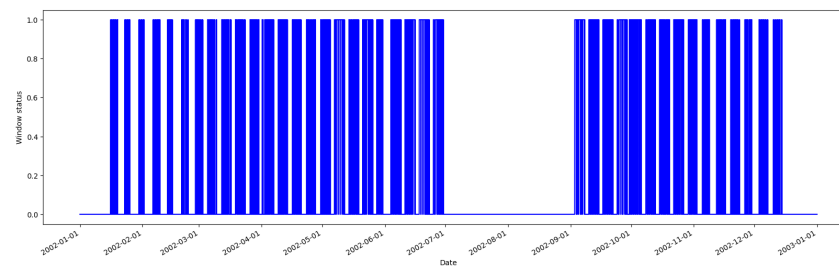


Figure 5.7 – Window status simulation case

5.2.2 Real case study

As mentioned in Chapter 1, one challenge we face is that the performance of activity detection can be affected by sensor placement. Therefore, only the optimal sensor (sensor 101) found in the Chapter 3 is used in this study. In addition to the ambient sensor, there are four window status sensors on each window and two door status sensors (see layout in Figure 5.8), a weather station, electricity consumption sensors, and a class schedule file. As the class schedule file may not accurately reflect reality due to random room entries and absences of students, so for the multi-activities detection, we will only use the simulation data to test our methodology.

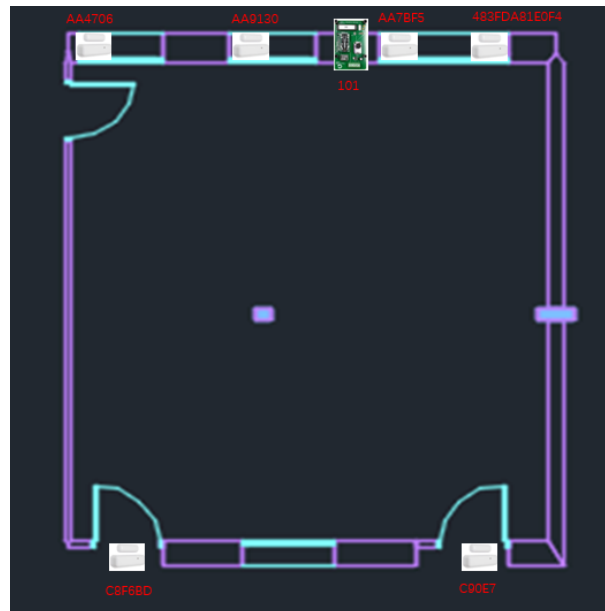


Figure 5.8 – Window door sensor and selected sensor

5.3 Self-learning supervised method

5.3.1 Background of self-learning supervised method

In order to understand self-supervised learning, remember that the main difference between supervised and unsupervised is whether the model requires manually labeled information during training. Typically, supervised learning is trained for a specific task using large manually labeled datasets. It not only relies heavily on expensive manual labeling, but also suffers from generalization errors. In order to solve this challenge, we aim

to find models that can learn more with fewer labels, fewer samples or fewer trials. As a promising candidate, self-supervised learning has received a lot of attention for its excellent data efficiency and generalization ability. Figure 5.9 vividly illustrates the differences between supervised, unsupervised, and self-supervised learning. In an autoencoder (un-

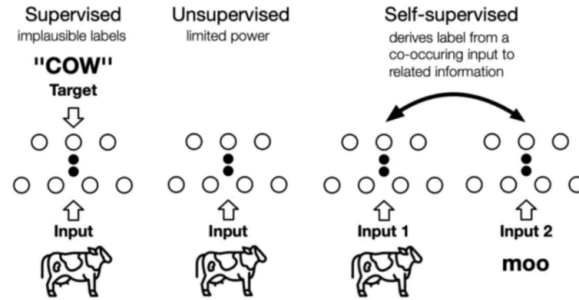


Figure 5.9 – Supervised VS unsupervised VS self supervised learning (Liu et al.2023)

supervised learning), the encoder maps the input samples to a latent/hidden Hermitian vector, and the decoder maps this latent/hidden vector back to the sample space. The goal is to achieve consistency between the input and output (ideally, lossless reconstruction), while the dimension of the latent/hidden vector is much smaller than that of the input samples. This achieves the dimensionality reduction objective. Utilizing the learned latent/hidden vectors simplifies tasks such as clustering and other downstream processes. The process of learning these latent/hidden vectors is known as Representation Learning. However, the encoding-decoding structure of autoencoders has limitations. Reconstruction losses based on individual data usually assume data independence, reducing their ability to model correlations or complex structures. In particular, the use of L1 (least absolute deviation) or L2 (least square errors) losses to measure the gap between inputs and outputs lacks semantic information. Overemphasis on individual data-level details neglects more important semantic features.

In the case of self-encoders, dimensionality reduction alone may not suffice. The goal of learning is not just dimensionality reduction but also capturing more semantic features, allowing the model to understand what the inputs actually are, and thus aiding downstream tasks; the main goal of self-supervised learning is to learn richer features representations.

Self-supervised learning mainly utilizes auxiliary tasks (pretext) to mine its own supervised information from large-scale unsupervised data. The network is trained with this

constructed supervised information so that it can learn representations that are valuable for downstream tasks.

However, self-supervised learning raises three fundamental questions: (i) How to perform representation learning on a large amount of unlabeled data? (ii) How to design effective auxiliary pretext tasks derived from the data itself? (iii) How to evaluate the effectiveness of the learned representations?

To address these questions, the ability of self-supervised learning is primarily assessed by the Pretrain-Finetune paradigm. The Pretrain-Finetune process is akin to supervised learning. First, a model is pre-trained on a large set of labeled data. Then, for a new downstream task, the pre-trained model's parameters are fine-tuned on the new labeled task to adapt it to the new objective. In self-supervised learning, the Pretrain-Finetune process involves training the network on a large amount of unlabeled data by constructing pseudo-labels through pretext tasks based on the data's inherent structure or characteristics. For example, in computer vision, transformations such as rotation, cropping, or flipping may be applied to images. Predicting attributes like the angle before and after rotation or the direction before and after flipping based on these transformations automatically generates labels. Training on these labels helps the model to understand the structure and features of the image and thus learn the computer vision task. Once a pre-trained model has been obtained, all that is required for new downstream tasks, as with supervised learning, is fine-tuning after migrating the learned parameters.

Self-supervised learning is a machine learning approach where models learn from unlabeled data through pretext tasks. For instance, in Natural Language Processing (NLP), it involves predicting surrounding words of a central word. In NLP, MASK LM training removes words from sentences, and models predict the missing words (Devlin et al. 2019). In image processing, the 'Jigsaw' method divides images into segments, and models predict their relative positions (Doersch et al. 2015). Advanced pretext tasks include inpainting, where models predict missing image parts (Deepak et al. 2016). Colorization tasks require predicting colors in grayscale images (Zhang et al. 2016). Split-Brain Autoencoders divide data into parts, encouraging models to understand semantic information (Zhang et al. 2017). More detail about these studies can be found in Appendix B.1

Self-supervised learning has seen success in computer vision and NLP but remains underutilized in building contexts like multi-activity detection. Saeed et al. (2019) used

wearable sensors to explore self-supervised learning, distinguishing human activities with limited labeled data. These methods show potential for enhancing transfer and semi-supervised learning. Inspired by these efforts, we investigate self-supervised learning’s suitability for multi-occupant behavior detection using only ambient sensor data for representation learning.

5.3.2 Methodology and result for the Multi-Occupant Behavior Self-Supervision Learning

5.3.2.1 Self-supervision learning methodology

In this section, we will apply the Multi-Occupant Behavior Self-Supervision Learning on the simulation dataset, aiming to classify four different scenarios: window open without occupants, window closed with occupants, window open with occupants presence, and window closed without occupants presence.

As the cornerstone of our self-supervised learning pipeline, we need to employ a pretext task that facilitates rich feature learning, data efficiency and transferability to downstream tasks. For this purpose, we utilize centered time shuffling. Sensor readings naturally follow a temporal order, and this temporal structure contains valuable information about the system’s dynamics. By shuffling the time indices and training the model to predict the original sequence, we force the model to learn the underlying temporal patterns and correlations between different sensors.

In the pretext phase, we employ the ANN method, which was introduced and elaborated upon in the previous chapter, to predict the correct temporal sequence. The same structure than in Chapter 4 is employed, consisting in one input layer with six nodes, following with hidden layer with twelve nodes, another hidden layer with eight nodes, and the output layer with one node for the temporal ordering of the data. This choice is motivated by the ANN’s exceptional performance seen in the previous chapter, where it effectively captured intricate underlying patterns. Computational time is of less concern during the pretext phase, allowing for the utilization of more complex models. Then, features will be extracted from the final layer of the ANN, which represent the learned representation of all features.

Subsequently, we proceed with transfer learning to adapt the learned representations for downstream tasks. Fine-tuning is performed based on the specific requirements of the downstream application, which, in our case, involves distinguishing among four different

scenarios. For this purpose, we employ the Histogram-Based Tree Learning XGBoost algorithm, as chosen in the previous chapter. The rationale behind this selection lies in XGBoost’s proven high accuracy and minimal computational time requirements, establishing a robust foundation for future building management control scenarios.

The steps are outlined below and illustrated in the flowchart of Figure 5.10:

1. Shuffling time and create labels. In this step, we will need shuffling time to create pretext, and creating pseudo labels, as following,

- **Data Preparation:** Start with a dataset D that contains sensor readings along with their timestamps.

$$D = \{(X_1, t_1), (X_2, t_2), \dots, (X_N, t_N)\} \quad (5.1)$$

- **Randomization:** Shuffle the dataset D to produce D' .

$$D' = \text{Shuffle}(D) = \{(X_{i_1}, t_{i_1}), (X_{i_2}, t_{i_2}), \dots, (X_{i_N}, t_{i_N})\} \quad (5.2)$$

- **Label Creation:** The shuffled sensor readings X become the features, and their original timestamps t become the labels.

$$\text{Features: } X' = [X_{i_1}, X_{i_2}, \dots, X_{i_N}] \quad (5.3)$$

$$\text{Labels: } T' = [t_{i_1}, t_{i_2}, \dots, t_{i_N}] \quad (5.4)$$

- **Data Splitting:** Divide the shuffled features X' and labels T' into training and test sets.

Then an ANN model F will be build and used to predict shuffled sensor readings X' to their corresponding original timestamps t . The model is parameterized by θ . The ANN structure will be same as the one in the last chapter, except the output will be timestamps.

$$F : X' \rightarrow \hat{t} \quad (5.5)$$

$$\hat{t} = F(X' | \theta) \quad (5.6)$$

Here, we could measure the Mean absolute error (MAE) as follows to assess the performance of the model F , by calculating the difference between predicted timestamps \hat{t} and the actual timestamps t .

$$J(\theta) = \min_{\theta} \frac{1}{N} \sum_{(X',t) \in D'} |F(X' | \theta) - t| \quad (5.7)$$

But this step is not necessary, since we will extract features from last layer before the output layer.

2. Transfer learning to downstream task

In this step, the learned representation must be transferred to new downstream task, after the pretext task. Following the pretext task, our trained ANN model F serves as a feature extractor. Specifically, we use the activations from the last fully connected layer as the feature vector for each data point.

$$\phi(X) = F_{\text{feat}}(X | \theta) \quad (5.8)$$

Given a dataset D of N data points, each data point transformed through F_{feat} becomes a row in the feature matrix A .

$$A = \begin{bmatrix} \phi(X_1) \\ \phi(X_2) \\ \vdots \\ \phi(X_N) \end{bmatrix} \quad (5.9)$$

3. Fine tuning for downstream task

Here, transferred learned representation will be used for downstream task, specifically for classifying four scenarios of occupants' activities. The feature matrix Φ will serve as the input, while the labels corresponding to the four scenarios will serve as the output.

$$\text{Training Data: } \{(\phi(x_1), y_1), (\phi(x_2), y_2), \dots, (\phi(x_N), y_N)\} \quad (5.10)$$

A small part of labeled data (5%, 10%, and 20% to the whole dataset) will be provided to the Histogram based tree learning XGBoost for training. Additionally, we will run a supervised method by XGBoost and ANN as a baseline for comparison. The Histogram based tree learning XGBoost model C is trained to map the feature vector $\phi(x)$ to the corresponding label y . The model is parameterized by ϕ .

$$C(\phi(x) | \phi) = y \tag{5.11}$$

The objective function $J(\phi)$ for the XGBoost is to minimize a loss function L that measures the difference between the predicted and actual labels like equation below.

$$J(\phi) = \min_{\phi} \sum_{i=1}^N L(C(\phi(x_i) | \phi), y_i) + \text{Regularization Term} \tag{5.12}$$

Whole steps can be seen in the Figure 5.10,

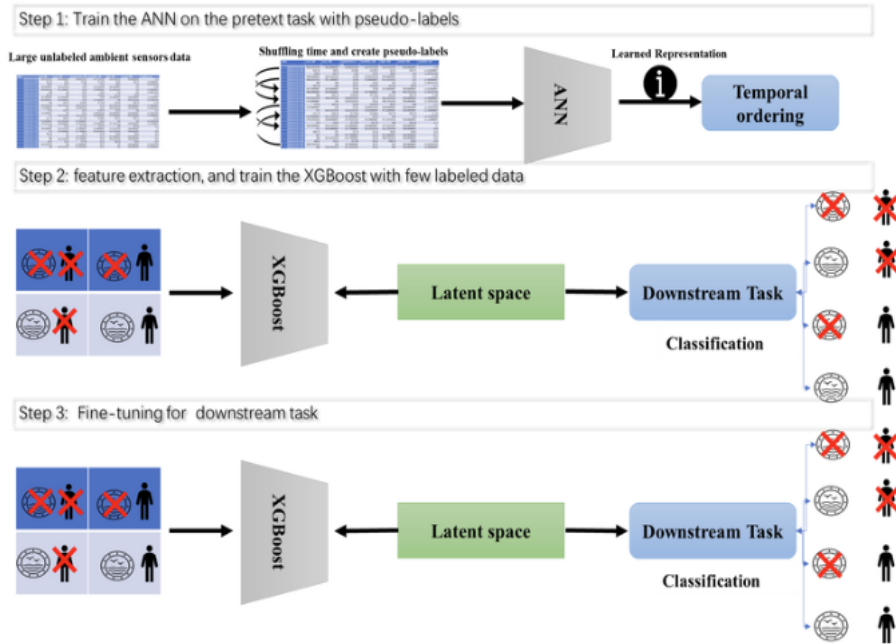


Figure 5.10 – Self supervision learning steps

5.3.2.2 First results obtained by crude self-learning

The essence of our experiment lies in transferring the learning capabilities from the pretext task to a downstream task. In this downstream task, the objective was to classify four distinct scenarios based on occupant activities and window positions. We fine-tuned the XGBoost model using a small fraction of labeled data, specifically 5%, 10%, and 20% of the entire dataset. The goal was to evaluate how well the model could generalize to this new task with minimal labeled data. By doing so, we aimed to demonstrate the efficiency of self-supervised learning in achieving high performance on the downstream task with limited labeled information. We also used supervised methods (ANN and XGBoost) as the baseline for the performance evaluation.

From the Table 5.1. we can see the accuracy (as defined in Chapter 1) for the self supervision learning with different percentages of label data and supervised method for comparison.

Table 5.1 – Algorithm Accuracy Comparison

Algorithm	Accuracy
5% label data self learning	81.05%
10% label data self learning	83.63%
20% label data self learning	86.85%
XGBoost	86.24%
ANN	84.90%

Figure 5.11 shows algorithms effectiveness on 5%, 10% and 20% labeled data.

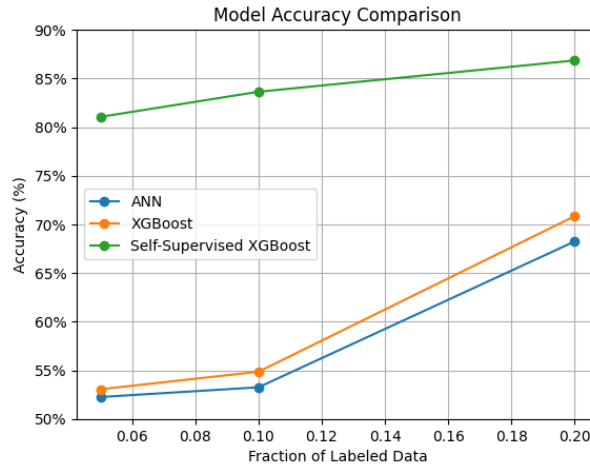


Figure 5.11 – Three algorithms with low percentage labeled data

Figure 5.12 displays the confusion matrix for the self-supervised learning approach with 20% labeled data.

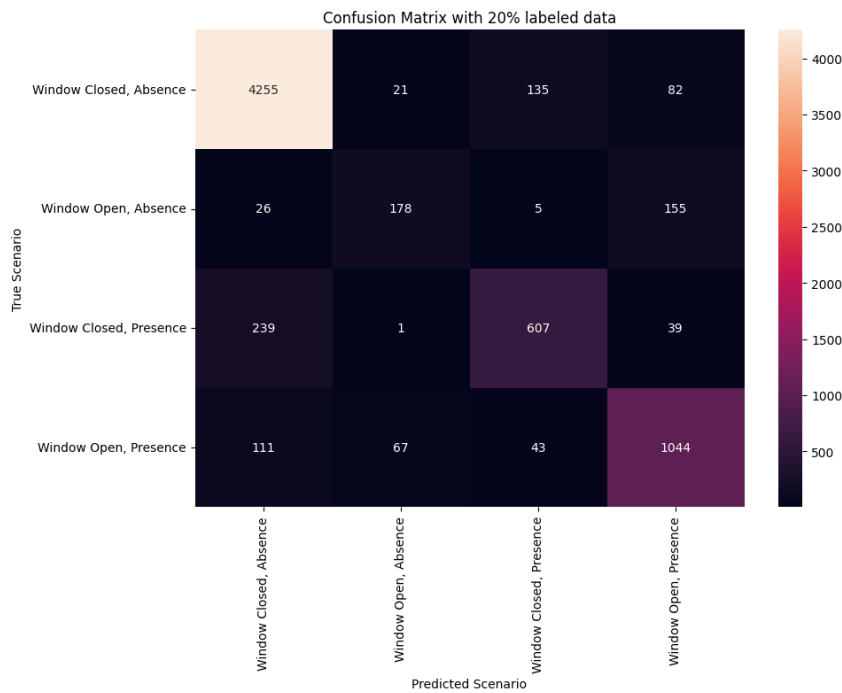


Figure 5.12 – Self supervision learning confusion matrix

Both the ANN and XGBoost models yielded robust results. Interestingly, when using

only 20% of labeled data, our self-supervised learning approach not only matched but also slightly outperformed the fully supervised models. This suggests that the features extracted through the pretext task are highly informative for the downstream task. The efficiency of our approach is likely bolstered by the inherent strength of the XGBoost model, which adeptly leverages the available features. In contrast, both XGBoost and ANN are trained based on 80% labeled data. This underscores the evident advantages of the self-learning algorithm, especially in scenarios with limited labeled data or when there is a need to uphold user privacy.

The confusion matrix confirms the model's effectiveness. A predominance of values along the diagonal underscores the model's commendable accuracy across varied scenarios. Notably, the model exhibits heightened precision when predicting scenarios of "window closed with no presence" and "window open with presence." This suggests that these situations exert a significant influence on the building's state, enabling the model to discern the underlying relationships more adeptly.

Our findings are consistent with existing research. For instance, Chen et al. (2020) demonstrated that their self-supervised learning algorithm, SimCLR, outperformed supervised methods in image recognition tasks. They further noted that increasing the number of parameters in the dataset could lead to even better performance. This supports the effectiveness and potential of self-supervised learning approaches in different fields.

5.4 Combination of DBSCAN, BN and Hotspot algorithm for multiactivities detection

In the present section, our main objective is now to develop a second method able to identify which activity causes a change of state in the ambient conditions. This method combines the Hotspot algorithm, DBSCAN, and a BN. The BN structure can be learned from historical sensor data and can be used to infer the probability of different activities based on new sensor data. As BN will infer from the found root causes, we have to introduce the BN before the root causes analysis. Finally, we will provide an overview of hybridized Hotspot and DBSCAN methods applied to our context.

5.4.1 BN in the perspective of root cause analysis

As mentioned in the introduction, evaluating the performance of unsupervised method poses a challenge. Furthermore, various activities, such as opening windows or people entering and exiting a room, can result in variations in multiple attributes (e.g., temperature, CO₂, etc.) within a building. Conversely, distinct attribute combinations can represent different combinations of activities. Therefore, we need a method capable of inferring the relationships between variables and activities. The BN offers a promising method. BNs have been applied in the building domain for diverse purposes, including energy consumption prediction, building control, and occupancy detection based on sensor data. These applications leverage the ability of BNs to model complex, multi-variable systems and make predictions under uncertainty.

5.4.1.1 General aspects and BNs steps

In the context of building activity detection, BNs can be employed to model the relationships between sensor data (e.g. temperature, light, occupancy) and activities (e.g. cooking, watching TV, sleeping). The structure of a BN is defined by its directed acyclic graph (DAG), which specifies these relationships between variables. The probability of each variable is represented by a conditional probability distribution (CPD), which is conditioned on the values of its parents in the DAG. Once the structure and CPDs of a BN are defined, the network can be used to make inferences about the variables.

The BN can perform various types of queries such as calculating the probability of a certain variable given the values of other variables or determining the most likely state of a variable given the values of other variables. This capability is particularly valuable when making predictions or decisions based on uncertain information.

The general steps for BN to perform inference can be summed up as follows:

1. **Define the problem:** Identify the relevant variables and their relationships based on expert experience. Determine the specific query you want to answer and the available evidence / knowledge.
2. **Model the BN:** Create a directed acyclic graph (DAG) representing the dependencies between the variables. Each node in the graph corresponds to a variable, and the directed edges indicate the causal relationships between these variables.
3. **Specify the conditional probability tables (CPTs):** For each variable, define a CPT that describes the probability distribution of that variable given its parents in

the graph. The CPTs represent the quantitative information about the relationships between the variables.

4. **Incorporate evidence:** Update the CPTs with any known evidence, which includes observed values of variables. This step is also referred as the conditioning step, where you condition the network on the observed evidence.
5. **Perform inference:** Use an inference algorithm to compute the posterior probabilities of the unobserved variables (or the variables of interest) given the provided evidence. Various algorithms can be employed for this step, including exact inference methods (e.g., variable elimination, clique tree propagation) and approximate inference methods (e.g., Markov Chain Monte Carlo, belief propagation).
6. **Interpret the results:** Analyze the inferred posterior probabilities to answer the query or make informed decisions. These posterior probabilities reflect the updated beliefs about the variables of interest after incorporating the evidence.

5.4.1.2 BN architecture design

For designing a BN, the most important aspect is building its structure or architecture, as it significantly influences the reliability of the inferences derived from the network. In the following, we will explore different methods for estimating the BN structure (domain knowledge, constraint-based MMHC, Tabu search, Chow-Liu tree, Hill-Climb). We will evaluate their performance based on the accuracy of inferred window opening using simulation data. Ultimately, we will compare the accuracy of different estimated structure to identify the most suitable one. The variables for simulation case and their designated letters will be introduced below,

Table 5.2 – Variables for the simulation case study

Variable Name	Symbol	Levels and Categories
WindowOpen	W	close:0 and open:1
Electricity	E	A, B, C
Solar	S	A, B, C
Heating	H	A, B, C
Indoor temperature	I	A, C
Occupancy	O	without:0 and with:1
Outdoor temperature	Ou	A, B, C

Each parameter, except for window status and occupancy, will be discretized using

the KBinsDiscretizer technique. This method divides the observed data range of each continuous variable into four equally spaced bins. These bins are then labeled with ordinal symbols from A to D. During the data transformation process, the value of each data point is evaluated to determine the bin to which it belongs. Once identified, the original value is substituted with the corresponding ordinal label—A, B, C, or D—for that bin. Converting the data in this manner facilitates its use in algorithms optimized for categorical input.

The BN structure estimation methods can be categorized into four different groups: Score-based structure estimation (hill climb/tabu search/chow-liu); Hybrid structure estimation (e.g. MMHC); Constraint-based structure estimation [e.g. Peter-Clart algorithm (PC)]; Domain knowledge. Each algorithm in each group will be introduced in Appendix B.2. Their performance will be compared in the next sections.

5.4.1.3 Comparison of the methods dedicated to the estimation of the BN structure

The summary result for the performance evaluation for each BN structure estimation algorithm is shown in Figure 5.13. The simulation datasheet will be divided into 75% training datasheet for the BN to learn the observed data (i.e. to estimate the Conditional Probability Tables (CPTs) for each node in the network). The remaining 25% of the test datasheet will be used to infer the window openings. After using a given estimation algorithm to initialize the BN structure, and this specifying parent-child relationships, the datasheet has been split. This has been done this by setting the observed variables and querying the "Window Open" variable. The inference gives a probability distribution for "Window Open". We took the value that corresponds to the highest probability as our "predicted" value for that instance. In the end, we compare the true detection with the ground truth value to calculate the accuracy.

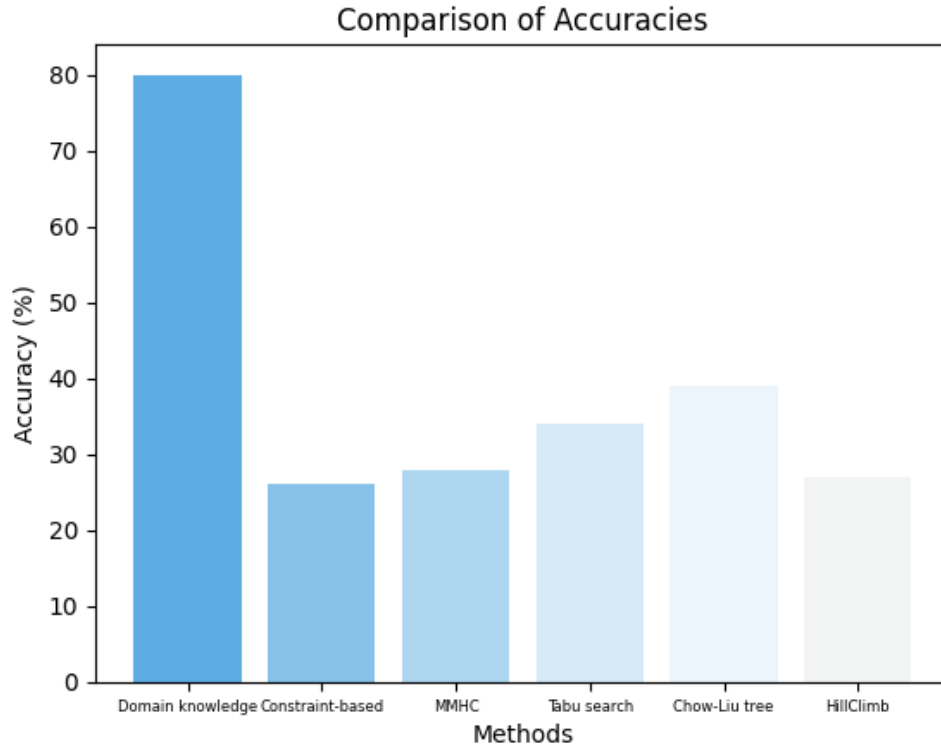


Figure 5.13 – BN structure comparison for simulated case study

In Figure 5.13, we can observe that all the methods do not achieve good performance besides the Domain knowledge method. In the Hill climbing, Tabu search, and MMHC algorithms, we noted that all the algorithms converge in the early stage, which means they are getting stuck in local optima. The search terminates in a suboptimal network structure, leading to the lower accuracy, even though the Tabu search aims to explore the search space more effectively by using a memory structure (the Tabu list) that stores recently visited solutions and prevents the search from returning to them. However, this does not necessarily ensure that the search will escape all local optima. If the true network is complex and not well-represented by the assumptions underlying these algorithms, the learned structure may deviate significantly from the true structure. For instance, Chow-Liu algorithm assumes a tree-structured network, while the PC algorithm assumes faithfulness. If the true structure does not meet these assumptions, the learned structure might be poor, which can explain the poor results for the Chow-Liu and PC algorithms.

To compare the speed of different algorithms for BN structure estimation, except domain knowledge, because its structure has been defined based on expert knowledge. Chow-Liu is the quickest because it scales linearly with the number of variables, especially

when the networks are tree-structured. Constraint-based estimation comes in second in terms of speed, being more efficient when dealing with a smaller set of variables. MMHC ranks as the third fastest ; it is a hybrid approach that merges the strengths of both constraint- based and score-based methods, offering a balanced trade-off between speed and accuracy. Tabu search is slower, coming in fourth place, largely due to the overhead of maintaining and updating the Tabu list. Hill climbing is the slowest among the five ; its time complexity is dependent on both the number of iterations needed for convergence and the size of the neighborhood it explores at each step.

Finally, relying on the domain knowledge, the BN structure for simulation case study is estimated and given in Figure 5.14 (the BN structure for real case study can be found in appendix).

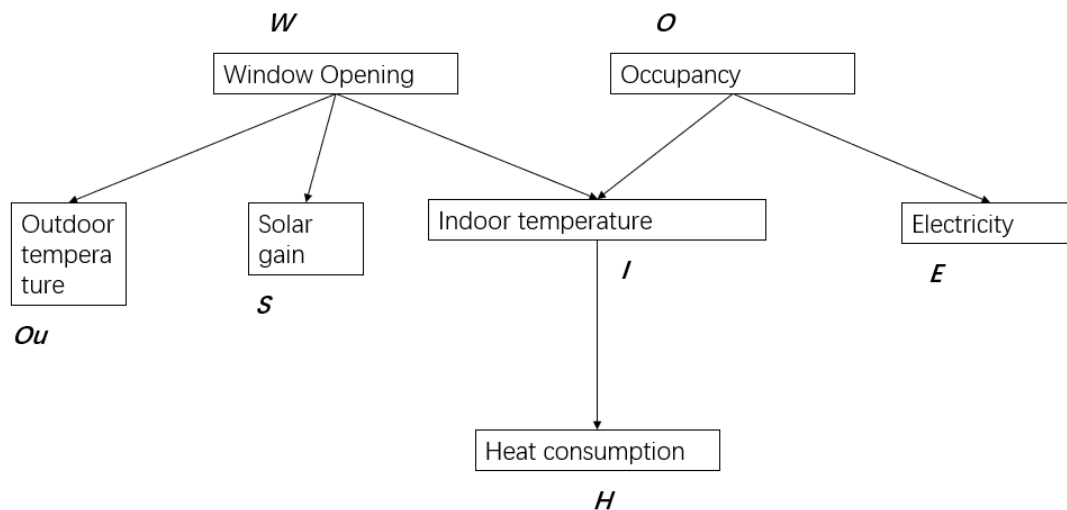


Figure 5.14 – BN structure of the simulation case study

As mentioned before, the direction of influence is indicated by the direction of the arrow between nodes. The arrows are from "parent nodes to "child nodes". For example in the simulation structure, Window Opening (W) directly influences its child nodes such as Outdoor temperature (Ou), Solar gain (S) and Indoor temperature (I). Additionally, Heat consumption (H) is a child node of the Indoor temperature (I).

In a BN, inference can flow in any direction—either from parent nodes to child nodes,

from child nodes to parent nodes, or even between nodes that do not have a direct parent-child relationship but are connected through other nodes. For example, when dealing with a parent node W and a child node O_u , given the conditional probability distribution $P(O_u|W)$, if we acquire new evidence E related to the child node C (e.g., $O_u = c$), we can apply Bayes' theorem to revise the belief (probability distribution) of the parent node W :

$$P(W|O_u = c) = \frac{P(O_u = c|W) \times P(W)}{P(O_u = c)} \quad (5.13)$$

5.4.1.4 Parameter learning

Parameter learning in a BN involves estimating the parameters of the CPDs (representing the probability of a random variable given its parent variables in the network). A BN encodes the joint probability distributions of the variables involved by factoring it into a product of CPDs. In the case of discrete variables, the CPDs can be represented as tables of probabilities. Let $\theta_{x_i|\mathbf{pa}(X_i)}$ denote the probability of $X_i = x_i$ given the specific instantiation $\mathbf{pa}(X_i)$ of its parents:

$$P(X_i = x_i | \mathbf{Pa}(X_i) = \mathbf{pa}(X_i)) = \theta_{x_i|\mathbf{pa}(X_i)} \quad (5.14)$$

The goal of parameter learning is to estimate the values of $\theta_{x_i|\mathbf{pa}(X_i)}$ for all i , x_i , and $\mathbf{pa}(X_i)$.

Each node of the discrete set of random variables $D = \{x_1, x_2, \dots, x_n\}$ is conditionally independent of its non-descendants, given its immediate parents. It is thus possible to factorize the conditional probability distributions over the set of variables by taking their product.

$$P(D) = \prod_{x_i \in D} P(x_i | pa(x_i)) = \prod_{x_i \in D} \theta_{x_i|pa(x_i)} \quad (5.15)$$

The result of this expression is sometimes denoted as Joint Probability Distribution (JPD).

Two popular methods for estimating parameters of statistical models or the table of probabilities exist: Maximum Likelihood Estimation (MLE) and Bayesian Estimation. Further details about these two methods and their application in the context of learning CPD parameters can be found in Appendix B.3.

In general, Maximum Likelihood Estimation (MLE) is more suitable for scenarios where simplicity and computational efficiency are prioritized, especially when there is no

available or relevant prior knowledge about the parameter values. On the other hand, Bayesian Estimation is more appropriate when it is essential to incorporate prior knowledge, or when a detailed probability distribution over the parameter values is crucial for decision-making or uncertainty quantification.

In our Bayesian estimation framework, we utilized the "BDeu" (Bayesian Dirichlet equivalent uniform) as our prior type. This represents a non-informative prior, suggesting that before any actual data observation, all potential outcomes or states of our variables are perceived as equally likely. The equivalent sample size (ESS) for this prior is set at 10. This can be practically interpreted as if we have introduced 10 pseudo-observations that resonate perfectly with a uniform belief. The comparison results between MLE and Bayesian estimation are presented in table 5.3.

With the Bayesian Estimation method, an accuracy of 85% was achieved, demonstrating the influence of the non-informative prior and the actual data on the posterior distribution.

Estimation Method	Accuracy (%)
Maximum Likelihood Estimation	78
Bayesian Estimation	85

Table 5.3 – Comparison of accuracies between Maximum Likelihood Estimation and Bayesian Estimation in simulation case study

Conclusively, by setting the ESS prior to 10, the Bayesian estimation tends to lean towards the prior information when estimating the parameters of the CPDs, rather than predominantly relying on the actual data. This approach might lead to improved generalization when juxtaposed with Maximum Likelihood Estimation. The heightened accuracy can be attributed to the model's proficiency in discerning the latent structure of the data by leveraging the prior information. This makes the model more resilient to noise and variations in the training dataset. Especially, Bayesian estimation proves beneficial when the training dataset is noisy or limited, as it curtails the model's tendency to overfit the training data, ensuring it grasps the core structure of the problem more efficiently. As a result, Bayesian estimation emerged as the chosen method for parameter estimation.

5.4.1.4.1 General aspects

The next step is to compute the JPD. To compute the JPD of a specific assignment of values to the variables in the BN, we follow these steps:

1. Give values to all variables in the network according to the specific assignment. For example, if we want to know the probability of "Window Open" ('W') being true when "Electricity" is 'A', "Solar" is 'B', and "Heating" is 'C',
2. For each node, look up the corresponding conditional probability given the values of its parents in the CPDs.
3. Multiply the conditional probabilities obtained in step 2 for all nodes in the network.

The result of this process is the joint probability of the specific assignment of values to the variables in the network.

Applied to our simulated case (shown in Figure 5.14), we can compute the joint probability of a specific assignment, say $P(W = w, E = e, O = o, S = s, H = h, I = i, Ou = u)$, following the steps outlined:

1. Assign values: $W = w, E = e, O = o, S = s, H = h, I = i, Ou = u$.
2. Look up the corresponding conditional probabilities for each node given the values of its parents:
 - $P(W = w)$
 - $P(E = e|O = o)$
 - $P(O = o)$
 - $P(S = s|W = w)$
 - $P(H = h|I = i)$
 - $P(I = i|W = w, O = o)$
 - $P(Ou = u|W = w)$
3. Multiply the conditional probabilities obtained in step 2 to get the joint probability:

$$\begin{aligned}
 P(W = w, E = e, O = o, S = s, H = h, I = i, Ou = u) &= P(W = w) \cdot P(E = e|O = o) \cdot \\
 &P(O = o) \cdot P(S = s|W = w) \cdot P(H = h|I = i) \cdot P(I = i|W = w, O = o) \cdot \\
 &P(Ou = u|W = w)
 \end{aligned}
 \tag{5.16}$$

By following the steps outlined, we can compute the joint probability of any specific assignment of values to the variables in the network. This JPD forms the foundation

for conducting inference tasks in BNs, such as calculating conditional probabilities and making predictions based on observed evidence.

5.4.1.4.2 Dealing with computational complexity

BNs often involve expensive computations, especially when dealing with large or high-dimensional datasets. For example, performing inference in a BN requires the computation of the posterior probabilities for certain variables given evidence from others. To mitigate the computational complexity, the Variable Elimination (VE) algorithm is widely employed in BNs. This algorithm leverages the conditional independence structure inherent in BNs, reducing the computational burden and enhancing the efficiency of inference. In this section, we will introduce the variable elimination algorithm and provide a simulated case study example.

The main idea behind the Variable Elimination algorithm is to iteratively eliminate variables from the JPD by summing them out. By doing so, we can compute the marginal probabilities of the remaining variables without the need to calculate the entire joint probability distribution. The algorithm consists of the following steps:

1. Identify the query variable (i.e. the variable for which we want to compute the probability distribution) and the evidence variables (i.e. variables with observed data, if any) from the BN.
2. Select an elimination order for the non-query and non-evidence variables. The choice of order significantly affect the efficiency of the Variable Elimination algorithm. A correct selection of the elimination order can drastically reduces the computational cost of inference. In here, we adopted the Min-Neighbors algorithm which prioritizes the elimination of variables that have the fewest neighbors first. A "neighbor" refers to any other variable (or node) directly connected to the current variable (or node) within the network. The reasoning is that variables with fewer neighbors contribute to smaller factor tables when summation is performed. Smaller factor tables require less computation to handle, which can make the overall inference process faster.
3. For each variable to be eliminated, compute a factor representing the product of all the factors involving that the variable and subsequently eliminate the variable from this product.

4. Multiply the remaining factors to derive the marginal probability distribution for the query variable.
5. If evidence variables are present, condition the marginal probability distribution based on the observed evidence.

Variables included in simulation case BN with the discretized level or categories data given in Table 5.2 (in the beginning of section 5.4.1.2), suppose we want to compute the probability distribution of the window status (W) given the following evidence:

$$S = C, \quad H = D, \quad I = A, \quad Ou = B \quad (5.17)$$

We can use the Variable Elimination algorithm to compute the conditional probability distribution $P(W|S = C, H = D, I = A, Ou = B)$. In this case, our query variable is W , and the evidence variables are S, H, I, Ou .

For the elimination order, we can choose the following ordering: S, H, I, Ou . Now, we will eliminate each variable according to this order:

- Eliminate S : Compute the factor $\phi_S(W) = \sum_S P(S = C|W)$, and sum out S .
- Eliminate H : Compute the factor $\phi_H(I) = \sum_H P(H = D|I)$, and sum out H .
- Eliminate I : Compute the factor $\phi_I(W, O) = \sum_I P(I = A|W, O) \times \phi_H(I)$, and sum out I .
- Eliminate Ou : Compute the factor $\phi_{Ou}(W) = \sum_{Ou} P(Ou = B|W)$, and sum out Ou .

We finally obtain the factor $\phi_{Ou}(W)$, which represents the unnormalized probability distribution of the window status (W) given the evidence. To normalize the distribution, we can compute the following:

$$P(W|S = C, H = D, I = A, Ou = B) = \frac{\phi_{Ou}(W)}{\sum_W \phi_{Ou}(W)} \quad (5.18)$$

This gives us the conditional probability distribution of the window status given the observed evidence.

5.4.1.4.3 Ending note

This section has outlined the procedure for constructing the BN structure. The next step involves utilizing inference within the BN to discern the activity responsible for

inducing a change in the ambient conditions, effectively pinpointing the root cause. We will now delve into a more detailed explanation of how root cause analysis operates.

5.4.2 Comparison of Root Cause Analysis algorithms and choice

Root Cause Analysis (RCA) is a systematic process used to identify the underlying causes of a problem or an incident. Instead of solely addressing the symptoms or immediate consequences, RCA seeks to uncover the fundamental issues that led to the problem in the first place.

RCA has found successful applications in various domains, including quality control issues (Burhabuddin et al.2022, Nanda et al. 2021, Sulistiyowati et al.2020), medical malpractice analysis (Slakey et al.2014), accident analysis (Arman et al.2022, Koo et al.2021), engineering failure analysis (Zwainy et al.2018, Martinez et al.2022). In recent years, root cause analysis has received a lot of attention and has been successfully applied to intelligent operation and maintenance in the community focused on anomaly detection and failure identification (Xu et al.2018, Zou et al.2022). We draw inspiration from the performance of RCA in anomaly detection, as occupant behaviors within buildings can be consider as root causes of anomalies. By employing root cause analysis to determine which attributes or combinations of attributes act as the root cause of an anomaly, we can detect and classify various behaviors without the need of labeled data. To the best of our knowledge, this marks the first instance of applying RCA to the detection of building occupant behaviors.

The main RCA algorithms documented in the literature can be categorized into two groups. The first group of RCA algorithms involves approaching the problem as an association rule mining task. Association rule mining is a method used to uncover patterns, correlations, associations or causal structures in datasets. Association rules are formulated as if/then statements, which help to find associations between independent data elements within the dataset. The efficacy of association rule mining heavily relies on selecting suitable thresholds. However, different datasets and fault cases may require different optimal threshold values, leading to less stable results. Because of these limitations, we have chosen not to use association rule mining and, instead, will employ heuristic search methods.

5.4.2.1 Heuristic search

The methods of this second group of RCA algorithms require defining an objective function, denoted as f : attribute combinations $\rightarrow \mathbb{R}$, which involves evaluating the root cause score given a combination (or a set) of attributes. Subsequently, the entire space is explored to identify the combination of attributes that maximizes the objective function and serves as the root cause. As most fault detection is oriented towards big data, the search space is often very large, necessitating the application of various heuristics and pruning techniques to reduce the search space. We compare five of the most significant heuristic search methods found in the literature. A short description is provided in Table 5.4 below. A detailed presentation is given in Appendix B.5.

Method	Description	Main Features and Focus	Applicability	Advantages/Strengths	Disadvantages/Weaknesses	Metrics	Comments
Adtributor (Ranjita et al., 2014)	Identifies root causes with a simplified approach.	Assumes only one attribute as the root cause.	Suitable for simple scenarios with single causes.	Simplicity	Limited to single-cause scenarios	Explanatory Power, Surprise	Useful for straightforward cases.
Recursive Adtributor (persson et al., 2018)	Extends Adtributor for more complex scenarios.	Iteratively applies Adtributor in a multidimensional context.	Handles scenarios with multiple contributing factors.	Handling multidimensional data	Limited by Adtributor's single-cause assumption	Explanatory Power, Surprise	Useful for scenarios with multiple factors.
iDice (Lin et al., 2016)	Targets anomalies in multidimensional time series data.	Uses pruning steps to narrow down search space and ranks root causes based on a score.	Effective for time series data, may not reduce complexity significantly.	Effective for time series data	May not significantly reduce complexity	Fisher Distance, Root Cause Evaluation Score	Suitable for time series data but complex.
Hotspot (Sun et al., 2017)	Focuses on performance data analysis.	Identifies significant factors affecting overall system performance.	Applicable in performance monitoring and optimization.	Performance-focused	Limited to performance analysis	Performance Metrics, Impact Analysis, others	Useful in optimizing system performance.
Squeeze (Li et al., 2019)	Extends Hotspot for basic and derived KPIs.	Applies heuristic strategies for faster root cause identification.	Suitable for a wide range of scenarios and KPI types.	Wide applicability	Heuristic methods may not cover all cases	Heuristic Strategies, Clustering, Anomaly Detection	Offers versatility and speed in RCA.

Table 5.4 – Summary of heuristic search methods

From the summary table of heuristic search methods (Table 5.4), it is evident that each algorithm has its unique strengths and weaknesses tailored to specific scenarios. In the search for an optimal Root Cause Analysis (RCA) algorithm for our study which aimed at deciphering the root causes of various activities inside a building, it is crucial that we align our choice with the specific constraints and requirements of the project. The building under investigation represents a complex system, thereby raising the possibility that the root causes of activities could be attributed to a singular or a multitude of factors. Given that the study's focus on multi-dimensional time-series data, our algorithm must also possess robust capabilities to effectively navigate and analyze such data.

The *Adtributor* method, though efficient in simple scenarios, is inherently designed for cases with only a singular root cause. This makes its applicability limited in our multifaceted setting. The *Recursive Adtributor*, while extending its capabilities for multidimensional contexts, remains bounded by the *Adtributor*'s foundational single-cause assumption.

iDice offers a tailored solution for anomalies in multi-dimensional time-series data. Yet, its inherent complexity and potential inefficiencies in managing large-scale data could pose challenges. The *Squeeze* method, with its versatility, primarily relies on heuristic methods. Such strategies might lead to overlooking specific potential root causes due to their nature.

In contrast, the *Hotspot* algorithm stands out as the most fitting choice. Its design, centered around performance data analysis, makes it capable at identifying significant performance influencing factors. Unlike other methods, *Hotspot* does not rely on rigid assumptions or aggressive data pruning. This adaptability, coupled with its proven proficiency in performance monitoring and optimization, ensures it is equipped to determine root causes even in vast search environments. Thus, making it ideally suited to our study's objective of identifying multiple activities inside buildings.

In light of these comparisons and the specific requirements of our study, the *Hotspot* algorithm is the method we have chosen for root cause analysis.

5.4.3 Development of a novel hybrid for occupant behavior analysis

The primary objective of the development presented in Chapter 5 is to conduct a comprehensive analysis of occupant behavior through the integration of fault detection and root cause analysis applied to time series data. This innovative approach is founded

on dividing the analysis problem into four distinct stages, each serving a specific purpose. These stages encompass the creation of a predictive model for heat consumption, unsupervised fault detection and classification, root cause determination, and the inference of window and occupancy status. Drawing upon our in-depth analysis and comparison of various methods conducted in the previous chapters and sections, we designed the hybrid approach detailed below.

The entire process is described in the flowchart in Figure 5.15. We use an ANN to predict heating (our KPI) through an ANN. We then use the DBSCAN-based anomaly detection to identify faults (occupant behavior) related to the window opening and occupancy status. Subsequently, the HotSpot algorithm identifies the root causes of these faults. Finally, a BN model is employed to infer the probabilities of different fault causes to determine different occupant behavior.

Therefore, the four underlisted steps are as follows:

1. Metamodeling Using Artificial Neural Network (ANN).

An ANN is trained using normal state simulated data features to create a regression model to predict ‘instantaneous heat consumption’. More details of this steps are described in §5.4.3.1

2. Unsupervised Fault Detection with DBSCAN

DBSCAN method is used to detect faults at different time points. It identifies instances / moments where the predictions significantly differ from the baseline simulation. It shows better and more stable performance for both simulated and real case studies in the last chapter. This step is detailed in §5.4.3.2.

3. Root Cause Analysis with Hotspot Algorithm

Once anomalies or faults are detected, the Hotspot algorithm comes into play. It analyzes the data around each detected fault point (occupant behavior) using a temporal window. This window includes historical data necessary for causal analysis. Since the simulation data has 5-minute time intervals, we use a temporal window that includes 30 minutes before and after the fault. We then use Monte Carlo Tree Search (MCTS) to find the best combination of dimensions that can explain the fault. This step is described in §5.4.3.3.

4. BN for Window and Occupancy Status Inference

Finally, a BN is utilized to infer the window and occupancy status. It does this by computing posterior probabilities based on the information gathered from the

previous steps. This step is detailed in §5.4.3.4.

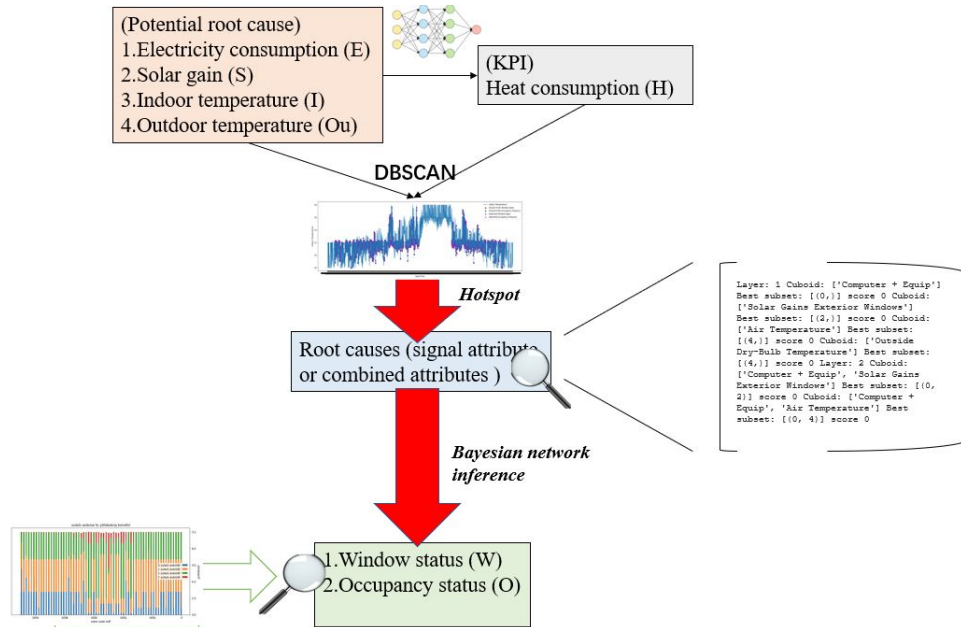


Figure 5.15 – Flow chart for multi activities detection

5.4.3.1 Step 1 : Metamodeling using Artificial neural network

In our context, KPIs essentially measure of how well a specific subset of the data explains the observed data. The predicted heating values serve as the "normal" baseline (no occupancy nor fault occurring).

An ANN model is constructed to predict the KPI using input features from the dataset. The model is designed as a feedforward neural network with two hidden layers since two hidden layers provide a balance between model complexity and the ability to learn non-linear patterns in the data. The network in Figure 5.16 shown the ANN structure for simulation with five inputs (Indoor temperature, Light, Electricity, Outdoor temperature, Solar gain) and heat consumption as the output:

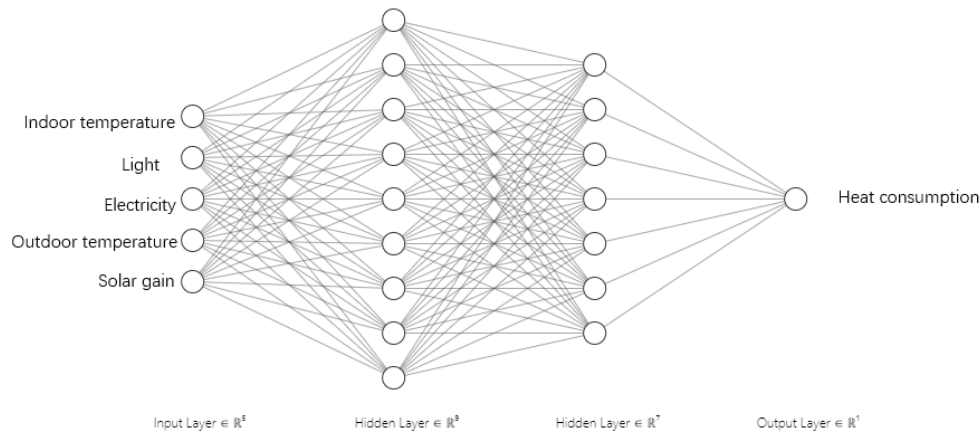


Figure 5.16 – ANN for simulation

The Rectified Linear Unit (ReLU) activation function is used in both input and hidden layers, while linear activation is used in the output layer. This choice aligns with the task at hand, which involves regression. ReLU introduces non-linearity in the model, enabling it to capture complex patterns efficiently. Additionally, it is computationally efficient and it can help mitigate the vanishing gradient problem during backpropagation.

Then model is trained using the ADaptive Moment estimation (Adam) optimizer (Chandra et al. 2021), enabling it to adjust the learning rate for each parameter individually, resulting in an adaptive learning rate. This facilitates faster convergence of the optimizer and requires less fine-tuning of the learning rate compared to other optimization algorithms like stochastic gradient descent (SGD).

During training, the mean squared error (MSE) is used as the loss function. Minimizing the MSE during training assists the model in learning the underlying patterns in the data. In the prediction process, only normal data (representing scenarios with no occupant behavior, i.e., no window openings and no occupancy presence) are used for training the ANN. This approach allows the ANN to learn the underlying patterns based on standard scenarios. Consequently, when a fault occurs, there will be a significant difference in heat consumption, as demonstrated by the prediction results depicted in Figure 5.17 below.

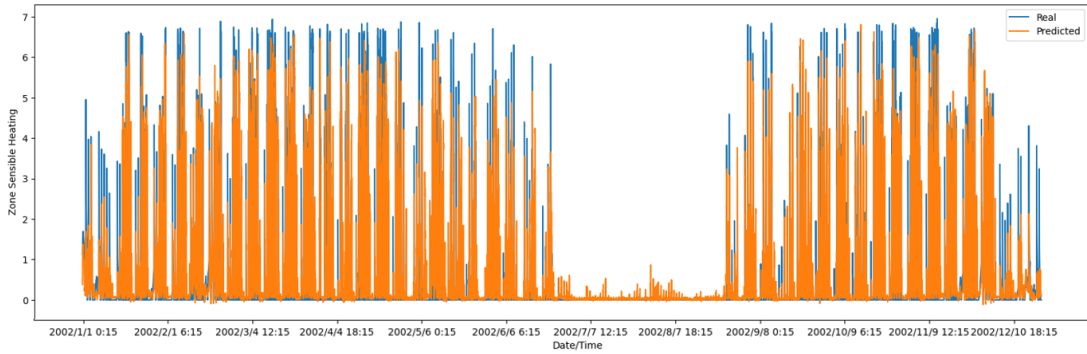


Figure 5.17 – ANN prediction for the simulation case

In the displayed figure, the Y-axis is the instantaneous heat consumption, and the X-axis is the Date time information, and the orange lines represent predicted values derived from standard data, while the blue lines show the actual values. Both sets of values exhibit a consistent trend across the year, suggesting that the prediction model effectively mirrors the underlying patterns in the real data. These cyclical patterns likely arise from daily fluctuations and seasonal influences. However, there are notable intervals where the predicted values significantly diverge from the real values, which might be attributed to anomalies in the data.

5.4.3.2 Step 2: Unsupervised Fault Detection with DBSCAN

The precomputed distance matrix DBSCAN method is used as in the chapter 4. The result to detect windows openings and presence of occupants can be seen in Tables 5.5 and 5.6 below,

Table 5.5 – Simulation window open confusion matrix

		Predicted	
		Closed	Open
Actual	Closed	20962	5974
	Open	115	8016

Table 5.6 – Simulation presence of occupancy confusion matrix

		Predicted	
		Absence	Presence
Actual	Absence	11446	12929
	Presence	3627	7038

From the two confusion matrices, we can notice that the accuracy is relatively high. However, it is noteworthy that DBSCAN frequently misclassified instances as openings or occupants. This misclassification is especially pronounced in the second case (presence), where the majority of actual absences are predicted as presence. However, in the Hotspot analysis, the primary focus is on identifying the root causes surrounding each detected activity, irrespective of the nature of these activities. The Hotspot approach does not hinge on the specific types of detected activities but instead leverages the ripple effect and a scoring mechanism to pinpoint the root cause. Thus, while there might be misclassifications, their influence on the final outcomes is minimal. Moreover, since the Hotspot method examines activities within a given time window, it inherently encompasses some undetected activities. This feature allows the Hotspot to effectively counteract and compensate for the limitations of DBSCAN, such as misclassification and detection lapses. In summary, numerous faults are detected and subsequently, the identified fault points for each simulation case will be inputted into the Hotspot algorithm to calculate the root cause during each fault occurrence.

5.4.3.3 Step 3: Root Cause Analysis with Hotspot algorithm

The purpose of this step is to identify the root causes within the window size of identified fault (Occupant behavior which was detected in Step 2). The output of this step will be used in the BN to infer the probability of each occupant behavior.

As previously mentioned in this chapter's introduction, the Hotspot algorithm is not well-suited for continuous time series data. Therefore, the first step will involve discretizing the continuous data into bins or discrete values. This discretization process enables spatial aggregation, making it feasible to identify patterns while reducing the influence of noise, emphasizing more significant patterns and relationships, and enhancing the speed of data analysis.

But, before proceeding with data discretization, it is crucial to consider the data's distribution characteristics to ensure that the discretization method captures the underlying data structure accurately. To accomplish this, we employ the Quantile-Quantile (Q-Q) plot, a graphical technique for comparing two probability distributions by plotting their quantiles against each other. Specifically, we utilize the Q-Q plot to verify if the data follows a normal distribution. In these plots, the red line represents the theoretical quantiles of the normal distribution.

The figures below illustrate the use of the Q-Q plot for simulation case parameters. If a parameter adheres to a normal distribution, the data points should align closely with the red line. However, in our case, it is evident that the data deviates from the red line. Notably, skewness can be observed in the indoor temperature, occupancy, outdoor temperature, and solar gain parameters, indicating a curve that diverges from the red line. Additionally, kurtosis is apparent in the heat consumption, window open, and electricity parameters, signifying a plot that rises more steeply or shallowly than the red line. These observations collectively suggest that the data does not conform to a normal distribution.

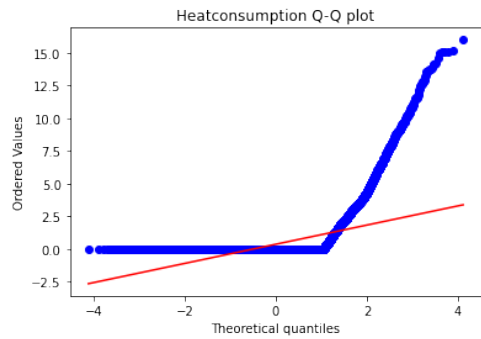


Figure 5.18 – Q-Q plot heat consumption

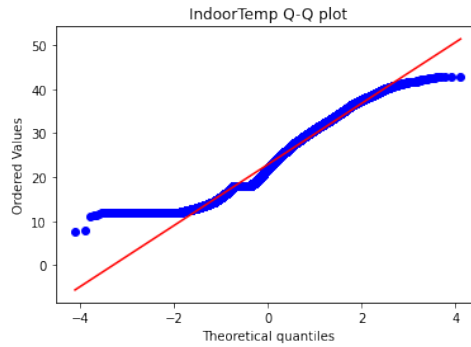


Figure 5.19 – Q-Q plot indoor temperature

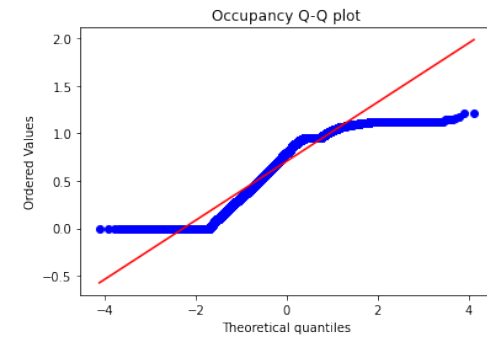


Figure 5.20 – Q-Q plot Occupancy

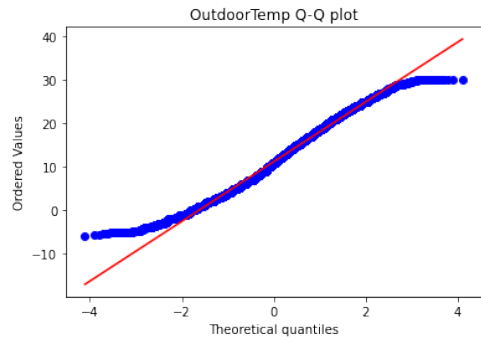


Figure 5.21 – Q-Q plot Outdoor temperature

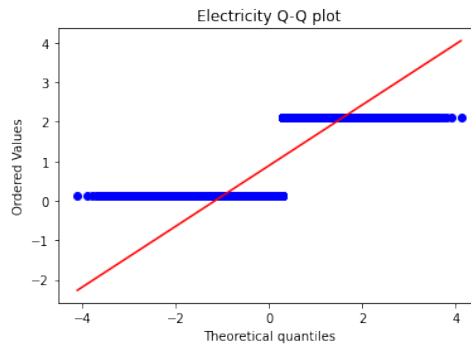


Figure 5.22 – Q-Q plot Electricity

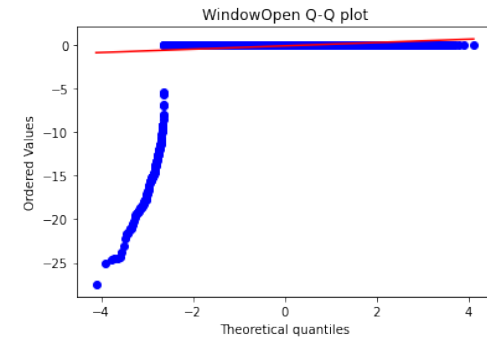


Figure 5.23 – Q-Q plot Window open

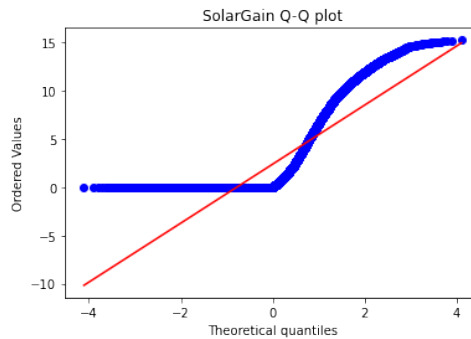


Figure 5.24 – Q-Q plot solar gain

With these observations in mind, we adopt the technique on which relies KBinsDiscretizer, an algorithm available as a class in the Scikit-learn library, in order to partition continuous features into discrete bins. One of its notable advantages is that it does not require the data to follow a normal distribution, making it suitable for our specific case. This technique allows us to specify the number of bins and select the binning strategy from options such as 'uniform', 'quantile', or 'kmeans'. For our purposes, we adapted the K-means clustering algorithm, which groups the data points into four bins. This adaptation takes into account the underlying data distribution and may yield better results, especially when dealing with time series data characterized by complex patterns or multiple clusters.

After completing the discretization process, the Hotspot algorithm can be used as an RCA method. As mentioned before, KPIs essentially measure of how well a specific subset of the data explains the observed data. In network operation and maintenance, internet companies often use KPIs to provide good quality of service. This also gives us inspiration that in buildings we can also find our own KPIs to evaluate what is happening in the building. For example, instantaneous heat production is usually caused by changes in events in the building (window opening or presence occupancy). In order to ensure the value of the set point, the control system will cause the instantaneous heat production to fluctuate. A KPI record can correspond to several attributes. For example, in Table 5.7, with the KPI (Heat) we have Solar (S), Temperature (I), Electricity (E), Outside (Ou), each attribute has a range of distinct values $S=\{s\}$, $I=\{i\}$, $E=\{e\}$, $Ou=\{ou\}$. And a vector of the distinct attribute value combination is called an element in this study as $e = (s, i, e, ou)$, where $(s \in S \text{ or } s = *)$, $(i \in I \text{ or } i = *)$, $(e \in E \text{ or } e = *)$, $(ou \in Ou \text{ or } ou = *)$, The asterisk serves as a wildcard. In this table, it shows one example of instantaneous heating consumption and its attributes as a function of fault duration, the attribute values are normalization value. The KPI values is record in every time interval (5 min in the study), for each distinct combination of the attribute values, e.g., (1.1151,-0.1197,1.5835,-1.0107). These elements are most fine-grained KPI records, because we are using the instantaneous heat production, then the KPI has the additive nature, can be naturally summed up into more coarse-grained KPIs, for example, all the KPI records with Solar=1.1151, Temperature=-0.1197 and Electricity=1.5835 regardless of Outside can be summed up into (1.1151, -0.1197, 1.5835, *). When we detected one fault, in the fault duration window, we can observe the increase and decrease, happens to a total KPI

(* , * , * , *). The elements most likely to influence the aggregate KPI during this period are considered the principal contributors to the anomaly.

Table 5.7 – Attributes and Heat value

Time	Solar(S)	Temperature(I)	Electricity(E)	Outside(Ou)	Heat
2002/1/24 11:15	1.1151	-0.1197	1.5835	-1.0107	0.318
2002/1/24 11:20	1.3643	-0.1197	1.5835	-0.9794	0.279
2002/1/24 11:25	1.5801	-0.1197	1.5835	-0.9401	0.239
2002/1/24 11:30	1.8002	-0.1812	1.5835	-0.9009	4.363
2002/1/24 11:35	1.9125	-0.1219	1.5835	-0.8617	0.327
2002/1/24 11:40	2.0176	-0.1197	1.5835	-0.8225	0.037
2002/1/24 11:45	2.1148	-0.1197	0.7807	-0.7872	0.161

Table 5.8 – Attributes and real/predicted Heat (Discretized)

Time	Electricity(E)	Solar(S)	Temperature(I)	Outside(Ou)	PredictedHeat	Heat
2002/1/24 11:15	B	B	C	A	0.783	0.318
2002/1/24 11:20	B	B	C	A	0.783	0.279
2002/1/24 11:25	B	B	C	A	0.633	0.239
2002/1/24 11:30	B	B	A	B	0.496	4.363
2002/1/24 11:35	B	B	B	B	0.326	0.327
2002/1/24 11:40	B	B	C	C	0.274	0.037
2002/1/24 11:45	A	B	C	C	0.352	0.161

Figure 5.25 depicts a 4-D cuboid data structure created from the simulation data.

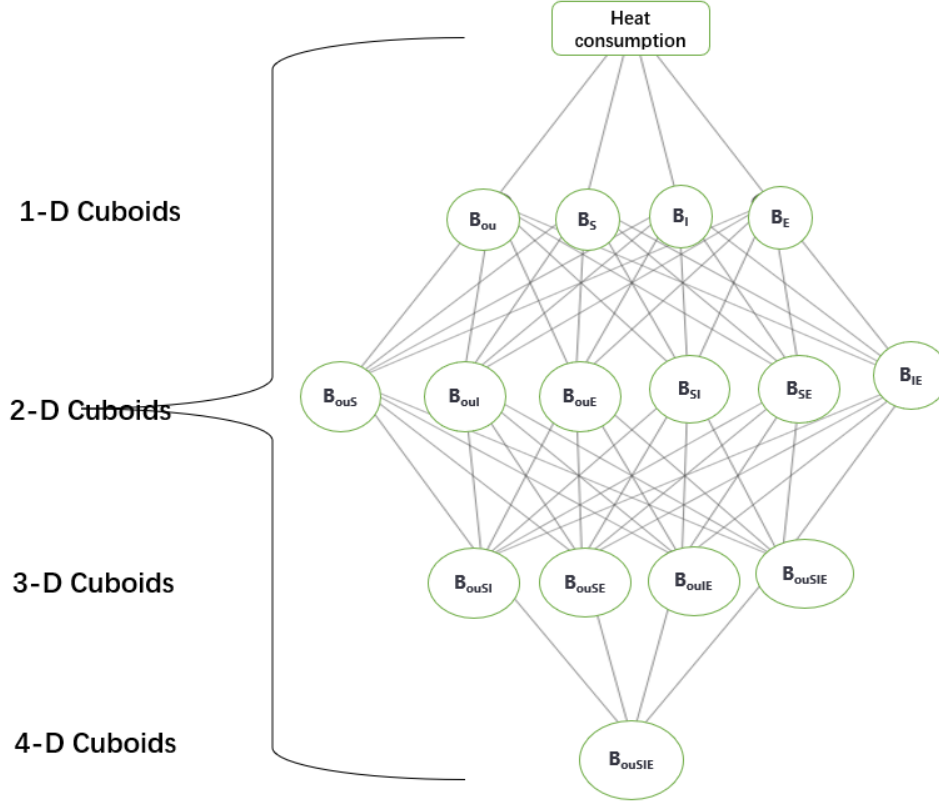


Figure 5.25 – 4-D cuboids

In each dimension, these cuboids group data points based on combinations of dimensions, such as pairs (e.g., electricity and temperature) and triples (e.g., electricity, temperature, and solar radiation). For example, B_S is a 1-d cuboid and B_{SI} is a 3-d cuboid. The element set of a cuboid B_S is denoted as $E(B_S)$, then we can have,

$$E(B_S) = \{e \mid e = (s, *, *, *), s \neq *\} \quad (5.19)$$

$$E(B_{SI}) = \{e \mid e = (s, i, *, *), s \neq *, i \neq *\} \quad (5.20)$$

In the study, all the cuboid and elements can be treat as a tree data structure to explore different combinations, e.g., B_S is the father cuboid of the B_{SI} , and their elements also have the father-and-child relationships. The gather of all these most fine-grained elements

are known as LEAF which is the ending nodes without children, it can be represent as:

$$\text{LEAF} = \{(s, i, e, ou) \mid s \neq *, i \neq *, e \neq *, ou \neq *\} \quad (5.21)$$

So, the other previous nodes (father nodes) which has one or more attribute value is asterisk, we can derive other element values by summing based on the elements in LEAF. As an illustration:

$$v(i = C, e = B) = 0.783 + 0.783 + 0.633 + 0.274 = 2.473 \quad (5.22)$$

$$v(i = C, *) = 0.783 + 0.783 + 0.633 + 0.274 + 0.352 = 2.825 \quad (5.23)$$

Hence, only the heat value (KPI) is directly record for the LEAF, If e' the descendant of e , then the aggregated heat value can be calculated as:

$$v(e) = \sum_{e' \in \text{Desc}'(e)} v(e') \quad (5.24)$$

In order to develop a metric which can be globally used to compare the root cause potential for all different element sets. Hotspot develop an idea called **potential score**, the idea behind is the KPI values of all descendant LEAF elements should also change with the root cause element. If an element is truly the root cause of a change, the effects of this change will be observed consistently in its descendant elements. This is because the descendants are, by definition, influenced by or dependent on their predecessor. If the actual changes in the descendants align well with the expected changes hypothesized by assuming a certain root cause, this serves as a confirmation (or at least a strong indication) that the root cause hypothesis is correct. Then, in here we have another definition **ripple effect** which refers to a conceptual model used to describe how changes in one element can cause subsequent changes in interconnected elements. This term is often used in various fields to denote a cascading sequence of events triggered by a single action.

The Table 5.9 show how to propagate the KPI change of a element to its descendant elements, in each cell, the first number is the actual Heating value $v(i, e)$, and the second number is the predicted Heating value $f(i, e)$. And the equation 5.25 show how the ripple effect works, Let x'_i denote the descendant elements of x . When the KPI value of x changes by $h(x)$ which is $h(x)=f(x)-v(x)$. x'_i will get its share of $h(x)$ according to the proportions of their forecast values. In the equation, the forecast/predicted

value is adjusted, contingent on the overall discrepancy observed, $h(x)$. The reduction is not uniform across predictions; instead, the discrepancy is distributed based on the relative size of each prediction. Consequently, larger predicted values undergo larger adjustments, and vice versa. Through this adjust, we can express the \vec{a} be the vector of $a(y_i)$, i.e., $\vec{a} = [a(y_1), a(y_2), a(y_3), \dots, a(y_n)]$, where n is the element count of $LEAF$, if $y_i \in LEAF$, it denote the newly deduced KPI values of an assumed root cause set S with $a(y_i)$. if $y_i \notin Desc'(S)$, $a(y_i) = f(y_i)$, the \vec{a} represents the adjusted values after considering root cause. This is an "ought-to-be" state, i.e., the state the system is expected to present if the activities were accurately identified and accounted for. Similarly, we can have the observation value vector $\vec{v} = [v(y_1), v(y_2), v(y_3), \dots, v(y_n)]$, and predicted/forecast value vector $\vec{f} = [f(y_1), f(y_2), f(y_3), \dots, f(y_n)]$. \vec{a} are calculated based on this distribution of change. Theoretically, if this distribution accurately reflects the actual situation, the adjusted values should be very close to the actual observed values.

$$v(x'_i) = f(x'_i) - h(x) \times \frac{f(x'_i)}{f(x)}, (f(x) \neq 0) \quad (5.25)$$

Table 5.9 – Discrepancies between actual and predicted KPI

v(i,e)→f(i,e)		E			
		A	B	C	*
I	A	0→0	0→0	0.16→0.35	0.16→0.35
	B	4.36→0.49	0.327→0.326	0.84→2.46	5.52→3.27
	*	4.36→0.49	0.327→0.326	1→2.81	5.68→3.62

In the Hotspot, it do not consider the direction whether to increase or decrease the values in \vec{f} ,so we can add a direction equation for ripple effect

$$direction = \text{sign}(\vec{v} - \vec{f}) \quad (5.26)$$

Here, the sign function returns -1 , 0 , or 1 , corresponding to \vec{f} being greater than, equal to, or less than \vec{v} , respectively.

$$\vec{a}[i] = \vec{f}[i] + direction \cdot \left(\frac{|\vec{f}[i] - \vec{v}[i]|}{\sum |\vec{f} - \vec{v}|} \right) \cdot h(x) \quad \text{for all } i \in \{1, \dots, n\} \quad (5.27)$$

With this, regardless of whether $h(x)$ is positive or negative, the adjustment will bring \vec{a} closer to \vec{v} . Then in the Table 5.9, the KPI of $(A, *)$ decreases from 4.36 ($v(A, *)$) to 0.49 ($f(A, *)$), given that $v(A, *)$ is aggregated from its descendant elements $v(A, A)$ and $v(A, B)$, these elements should exhibit corresponding changes. Let's define this change value as $h(A, *) = 3.87$, then the ripple effect of element (A, B) is calculated as

$$v(A, B)f(A, B) + h(A, *) \times \frac{f(A, B)}{f(A, *)} = 0.49 + 3.87 * \frac{0.49}{0.49} = 4.36$$

Then we can have the potential score which will be used to compare the root cause potential for all different element sets. The equation as follow:

$$\text{Potential Score} = \max \left(1 - \frac{d(\vec{v}, \vec{a})}{d(\vec{v}, \vec{f})}, 0 \right) \quad (5.28)$$

where $d(\vec{u}, \vec{w})$ represents the Euclidean distance of the vectors \vec{u} and \vec{w} :

$$d(\vec{u}, \vec{w}) = \sqrt{\sum_i (u_i - w_i)^2}. \quad (5.29)$$

In the equation 5.28, the $d(\vec{v}, \vec{a})$ measures the discrepancy or distance between the actual observed values and these inferred values. A larger distance indicates a significant deviation between the actual observed values and the expected state based on activity, suggesting that there may be other factors in the system not explained by the current activity. A smaller distance indicates that the actual observed values are close to the expected values considering anomaly adjustments, suggesting that the influence of the activity has been well captured and explained in \vec{a} .

And $d(\vec{v}, \vec{f})$ represents the distance between the actual observed values and the expected values under normal operating conditions. The larger this distance, the further the actual observed values deviate from the normal expected state.

When considering the ratio $\frac{d(\vec{v}, \vec{a})}{d(\vec{v}, \vec{f})}$, we are essentially comparing the degree of deviation of the actual observed data relative to these two expected states. If this ratio is less than 1, it means that the actual observed values are closer to the state considering the activity (\vec{a}), indicating that the current activity explanation is more consistent with the observed data, and the activity influence may be the primary factor for the deviation. If this ratio is close to or greater than 1, it means that the degree of deviation of the actual observed values

from the normal expected state (\vec{f}) is greater than or equal to the degree of deviation from the activity state (\vec{a}), suggesting that there may be other root cause affecting the data. Therefore, this ratio provides a method for assessing whether the current activity sufficiently explains the observed data changes, or whether there are other potential factors that need further exploration. In this way, it helps guide response measures, whether further investigation is needed, and the accuracy of predictive models. In order to make the search process more efficient, HotSpot uses a method called hierarchical pruning. This means that after it looks at the simpler layers, it removes some elements from the more complex layers that probably didn't cause the problem. This is because if a parent element is not likely involved in the problem, then each of the children elements is unlikely to be a root cause element, and thus can be pruned. This idea is similar to a known strategy in Association Rule Mining. The method is called "hierarchical" because it uses the layer information to decide.

The algorithm will continue to calculate the potential scores for each element and each cuboid until the maximum potential score is reached. At this point, the root cause for the given fault time duration is identified. The algorithm will then proceed to repeat this process for all fault points detected by DBSCAN, ensuring that each fault point is addressed. As mentioned before, the heating serves as a KPI in the HotSpot method is used to measure the significance of the deviation between observed and expected values for a particular attribute combination (referred to as an "element"). By isolating the root cause, one can take targeted actions to address the issue, rather than treating symptoms or secondary effects.

After finding the root causes, we will combine them with a BN in the next step to infer the probability of each occupant behavior.

5.4.3.4 Step 4: BN for Window and Occupancy Status

Figures 5.26 to 5.29 depict the results of applying the BN to the training data. It turns out that the BN with our structure (described in §5.4.1.3.) is highly effective at capturing the relationships and dependencies among the variables, leading to reliable predictions or estimations. In Figure 5.26 and Figure 5.28 respectively, we can observe the inferred probability of occupancy status (and window status, respectively), Orange lines indicate the inferred probability of occupant presence (and windows openings, respectively), while blue lines indicate the inferred probability of occupant absence (and closed windows, respectively). In Figure 5.27 (Figure 5.29 respectively), the BN selects high probability

as the true detection and compares it with the ground truth. In the Figure 5.30, it shows extracted one day window status and occupancy status inference with the ground truth. The blue round shows the window status 0 inference, and yellow round shows the window status 1 inference, green round shows the occupancy status 1 inference, red round shows the occupancy status 0 inference, and black cross shows the ground truth for window status, and red cross shows the ground truth for occupancy status, for the ground truth 0 means close or no presence, 1 means open or presence. In the window status, if the blue round (window status 0) reach a higher probability than the yellow round (window status 1), then we can consider the window status is 0, and vice versa. It also applies to occupancy status as well. We also can notice that most of points are detected correctly.

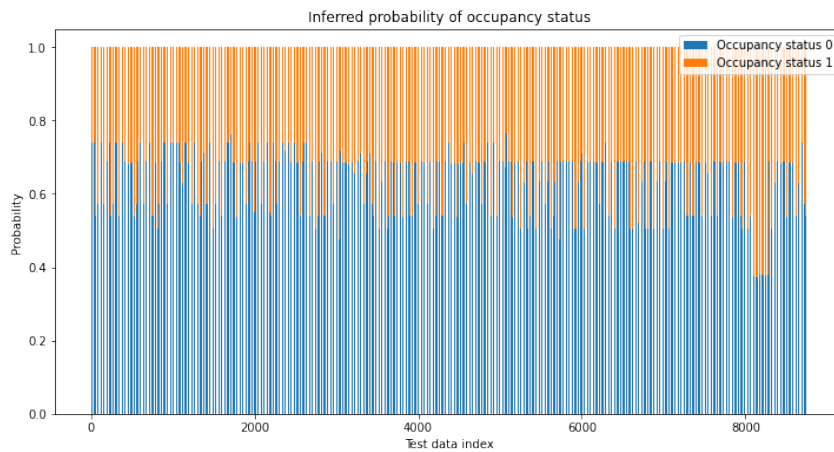


Figure 5.26 – BN for occupancy status inference

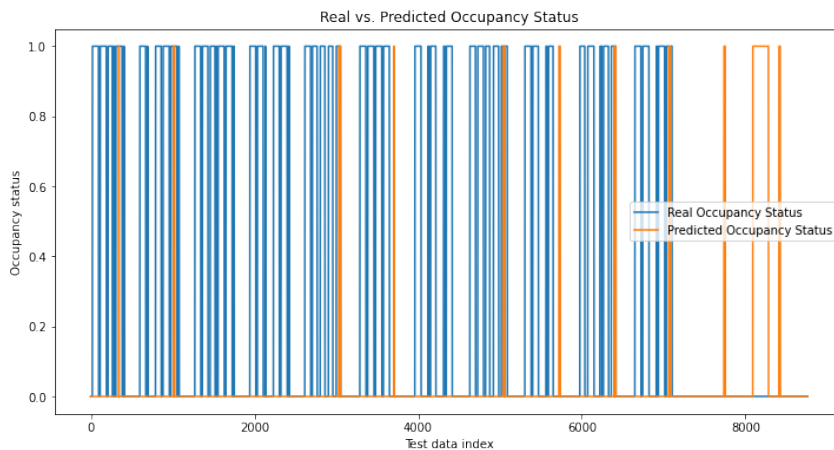


Figure 5.27 – Occupancy status real VS inference

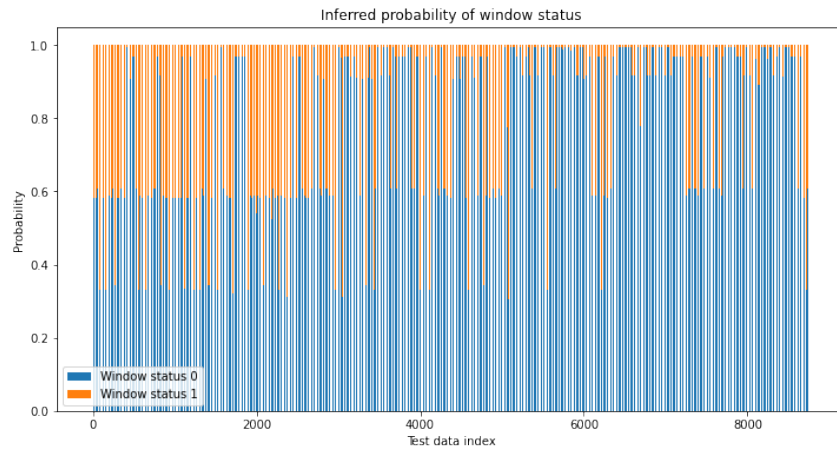


Figure 5.28 – BN for window status inference

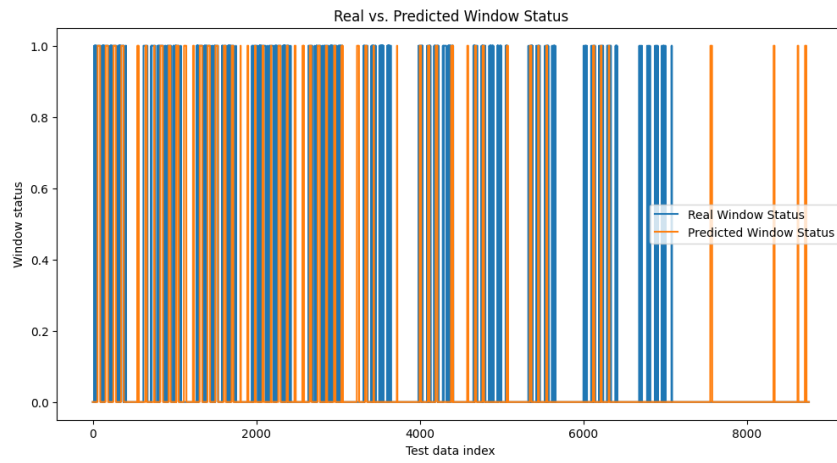


Figure 5.29 – Window status real VS inference

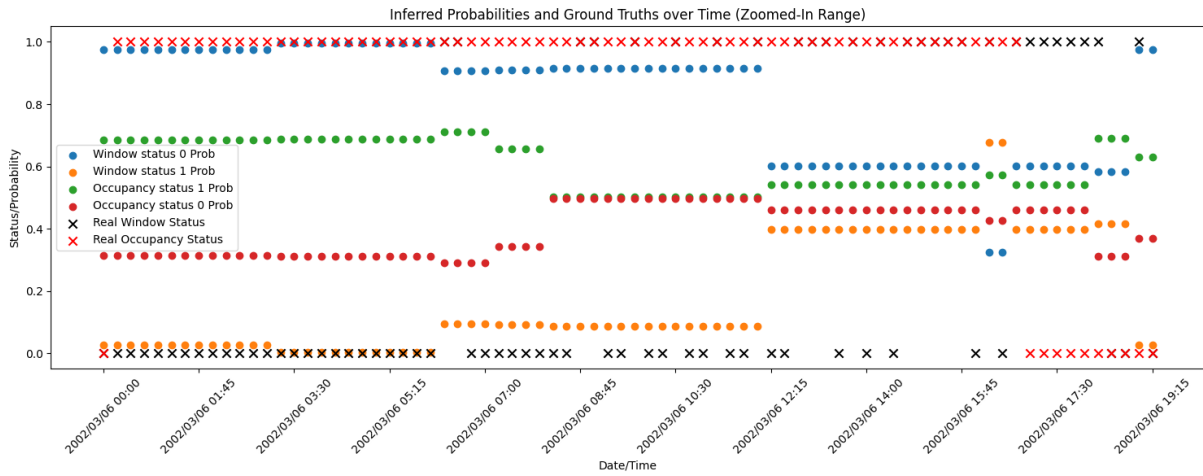


Figure 5.30 – Extracted one day BN inference

Particularly, the inference of window status in simulation data achieves a 85 % accuracy, which can be attributed to the correct network structure. This enables the network to make reliable inferences by accurately propagating the evidence through the nodes. Moreover, the appropriate discretization strategy allows the network to effectively represent the relationships between variables.

Another key factor contributing to the model’s accuracy is the choice of a suitable parameter estimation algorithm. The selection of the weight assigned to the prior distribution and observed data is crucial for achieving high model accuracy. However, the inference of occupancy status yields a less impressive result, with an accuracy of 70%. This may be due to the weak correlation between occupancy status and other variables, making it difficult for the model to learn the underlying relationships between them. One possible solution to improve the occupancy status inference is to generate more correlated data with occupancy status, which would enable the model to better capture the relationships and ultimately enhance its performance.

5.4.3.5 Results for our Integrated Behavior Analysis and Root Cause Detection hybrid method

To remind the whole concept of our approach, the ANN will be used to predict the instantaneous heating to serve as an auxiliary KPI in the HotSpot algorithm. We trained the ANN using all normal data (no window open and no occupant presence), and then the ANN predicted the instantaneous heating for the entire dataset. This predicted instantaneous heating can be considered as a baseline for when the building has no faults. Then,

in the HotSpot algorithm, we can observe variations in the real instantaneous heating because both of window open and occupancy presence lead to changes in the instantaneous heating. The results for the ANN are shown in Figure 5.17 (§5.4.3.1).

Secondly, the speeded-up DBSCAN will be applied to the dataset to detect faults (window open and occupant presence). There are a total of 12576 faults, as shown in Figure 5.31. The values represent the Heating consumption for the entire year, and the red spots indicate the detected faults.

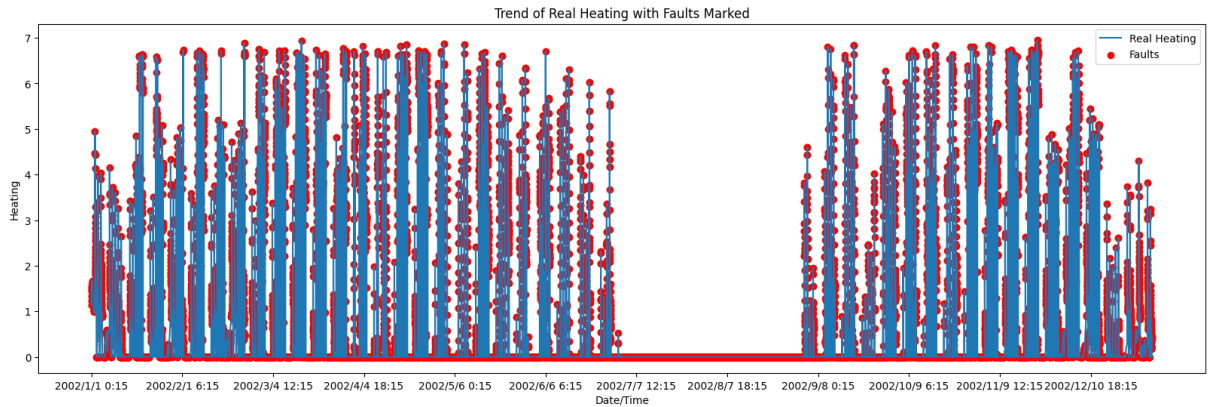


Figure 5.31 – Detected faults

Thirdly, root causes will be identified at each fault point within ± 30 min period of the fault. The HotSpot algorithm will calculate the root causes based on the predicted KPI, which is calculated from the ANN with ripple effect and potential scores. The Detailed steps can be found in section §5.4.3.3. Afterward, we can perform Hotspot RCA method for each fault window as shown in Table 5.10 below, each line indicate the found root cause parameter in each fault index, corresponding value, and calculated potential score. For example, in the fault index 14, the root cause is indoor temperature, it actually might imply that temperature is a key indicator or signal that the system's state has changed like the environmental changes, rather than the direct cause of the activities. And in mathematical terms, as temperature is a root cause, its changes are likely to have strong correlations with changes in all its descendants. This implies that when a change in temperature occurs, we can anticipate corresponding changes in these downstream variables. And we also can notice, there is root cause like solar radiation which is typically independent of direct human action within a facility or an immediate environment, while even it not directly altered by human behavior, solar radiation can still have a significant impact on internal conditions. For instance, a rise in solar radiation could lead to an

increase in room temperature, potentially triggering anomalies.

Fault Index	Cuboid	Value(discretize)	Score
11	Solar	0.1	0.433417
12	Electricity	1.58	0.404714
14	Indoor temperature	-1.79	0.425119
15	Indoor temperature	-1.79	0.423188
16	Solar	0.61	0.522567
18	Solar	-0.31	0.696077
19	Solar	-0.31	0.691519
etc.	etc.	etc.	etc.

Table 5.10 – Root Causes

Fourthly, the identified root causes will be used as evidence to input into the BN (structure show in Figure 5.14) for inferring the probability of the window opening and occupancy. The results can be seen in Figure 5.32. Many points are stacked together because: (i) there is a large number of faults; (ii) each fault will have a window of ± 30 min; (iii) and for each fault point, BN will give two inferences (one is the probability of window open, and the other is the probability of presence occupant). It should be noted that because the instantaneous heat is used as a KPI, faults occurring in seasons without heat consumption are discarded, resulting in gaps in the picture. Another enlarged picture at a specific time frame can be seen in Figure 5.33.

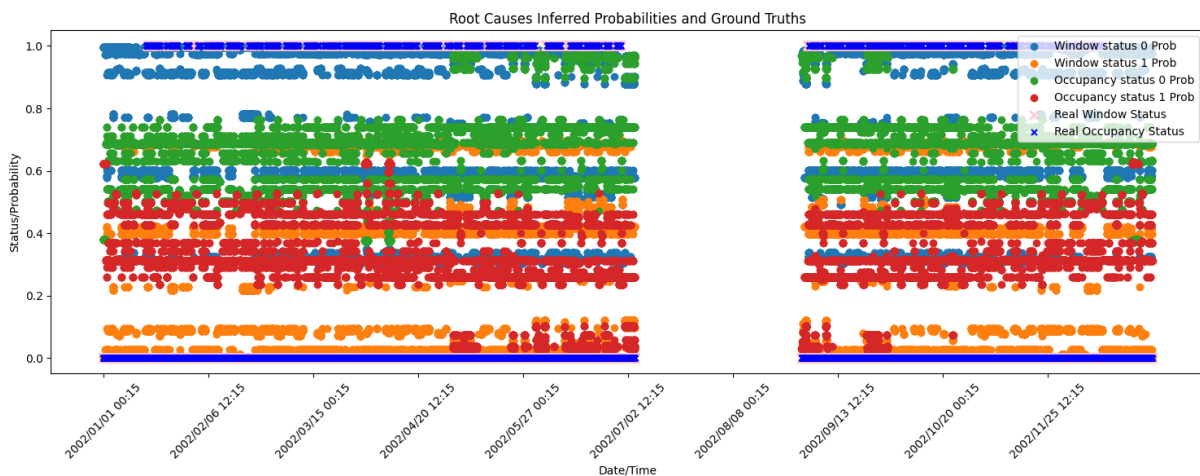


Figure 5.32 – Inference probability VS ground truth

From the Figure 5.33, blue and yellow round represent the window status inferred probability for 0 or 1, green and red round represent the occupancy status inferred probability for 0 or 1, and black cross indicate real window status, red cross indicate the occupancy status. If the probability of the blue round is bigger than the yellow round, we will consider the window status is close (0) as the true detection. We can notice that, the modal is good to the detect the window close status, but for the two inferred values (window open and presence of occupant) in each fault window, if both values exceed 60%, we consider them to have occurred simultaneously. Because it is not possible to correctly predict the state of occupancy, this also leads to inaccuracies in the classification detection of four types of activities (open and presence, close and presence, close and no presence, open and no presence).

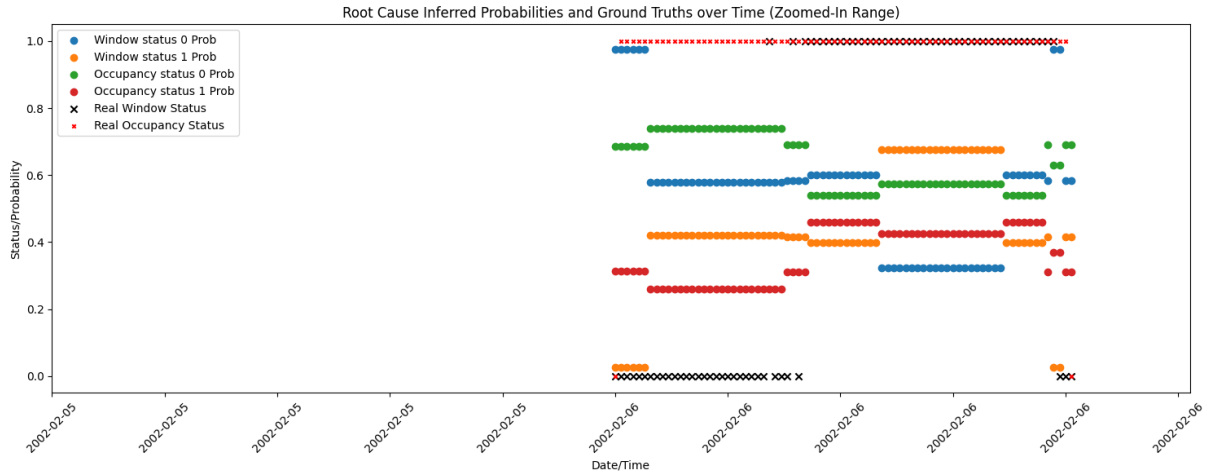


Figure 5.33 – Zoom in inference probability

Then; the final results will be compared with the ground truth and provide the confusion matrix below.

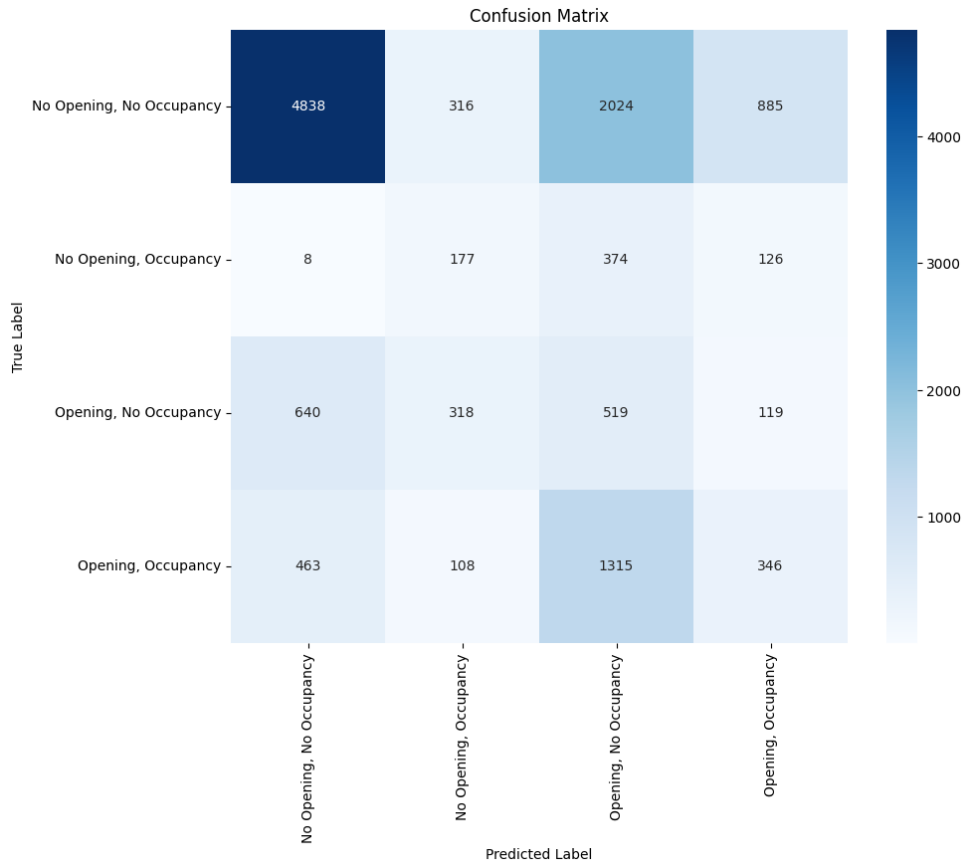


Figure 5.34 – Confusion matrix for the ANN-DBSCAN-HOTSPOT-BN

The confusion matrix offers a comprehensive overview of the classification model’s performance across distinct scenarios. Diagonal values in a confusion matrix represent the True Positives for each scenario, indicating instances where the model’s predictions align with the actual data. However, in this confusion matrix, not all diagonal values are maximized, highlighting areas where prediction accuracy is compromised. And the accuracy of the this ANN-DBSCAN-HOTSPOT-BN method can be calculated as 45%. This outcome is less than ideal, indicating that the model under unsupervised conditions lacks the ability to accurately discern different behaviors, but discerning different behaviors under unsupervised conditions is inherently a very challenging task. How to better and more appropriately select the root cause and how to choose the right attributes to input into RCA will be key to advancing the model

In the context of building environments, the indoor system exhibits intricate complexities. Various parameters undergo fluctuations in response to alterations in the building state, intertwining the real root causes with a multitude of factors. Given this intricate web

of interactions, pinpointing the authentic root causes emerges as a formidable challenge.

Our proposed model embodies this complexity, operating as an intricate system built upon the synergistic interplay of neural network predictions, unsupervised detection, Root Cause Analysis (RCA), and BNs. Within this delicate framework, even minor deviations can have cascading impacts, potentially undermining the entire system's efficacy.

The confusion matrix offers critical insights into the model's performance. we can notice the model's challenges in discerning between diverse behaviors. Given our stringent criteria, necessitating both inferred values (Window Open status and Occupant Presence) to surpass the 60% threshold, the model grapples notably with predictions pertaining to window openings and the presence of occupants. Despite the multifaceted challenges inherent in such a complex scenario, this model can still provide considerable successful detection, so it provides a good direction for future research.

Future research should delve into the intricacies of detection within such multifarious scenarios. Identifying wholly accurate root causes remains paramount. Additionally, determining suitable thresholds from BNs to accurately discern multiple concurrent activities will be instrumental in refining the model's precision. Another key area for improvement is the choice of root cause analysis methods. Unlike in web operations and maintenance, the original application domain for Hotspot, not all features in a building context can be directly aggregated. Therefore, finding a root cause analysis method that is better tailored to building environments could be crucial for improving performance.

5.5 Conclusion of the chapter

In this chapter, we first introduced a self-supervised learning approach to understand and model multi- occupant behaviors. We introduced a novel self-supervised learning approach, leveraging time shuffling as a pretext task for an Artificial Neural Network (ANN). Through transfer learning, we harnessed the power of this ANN for downstream tasks focused on detecting various occupant behaviors. Remarkably, our self-learning algorithm achieved an impressive accuracy rate of 86.85%, surpassing the performance of traditional supervised methods.

Next, we focused on the potential of BNs. We established a comprehensive benchmark to evaluate different methods for estimating the structure of BNs. Our analysis revealed that, particularly for complex BN structures, incorporating domain knowledge appears to yield the most effective results, as alternative methods often found themselves trapped in

local optima. We also explored parameter learning and variable elimination techniques, ultimately presenting the finalized structure of our BN. Subsequently, we proposed a method for identifying root causes underlying various occupant behaviors. We initially selected 'instantaneous heat consumption' as the KPI and employed an ANN to forecast this KPI under normal conditions. The precomputed distance matrix using DBSCAN method—chosen based on our findings in Chapter 4 allowed us to detect individual occupant behaviors. Following this, the Hotspot algorithm was employed to identify the root causes associated with each detected behavior. These root causes were then incorporated into the BN for further inference. While the results from the ANN-DBSCAN-HOTSPOT-BN algorithm indicate a certain level of proficiency in detecting different occupant behaviors, it is evident that there is substantial room for improvement.

From the observed results, the self-learning algorithm significantly outperforms the ANN-DBSCAN-HOTSPOT-BN method in both detecting single activities and distinguishing among multiple activities. Notably, the self-learning algorithm achieves this superior performance using only 5% of labeled data, aligning well with our objective of preserving user privacy. On the other hand, while the ANN-DBSCAN-HOTSPOT-BN approach is a multi-activities detection that moves towards unsupervised learning, its potential and relevance in this domain make it an avenue worth exploring in depth.

Future work could involve optimizing the root cause analysis methods to better align with the specific nuances of building environments. Additionally, further refinement of the BN structure and parameter learning techniques could enhance predictive accuracy. Ultimately, this chapter presents a multi-faceted approach to understanding and detecting occupant behaviors. These insights carry the potential to improve energy efficiency and occupant well-being, marking a significant contribution and laying the foundation for future advancements in the field of building management.

CONCLUSIONS AND PERSPECTIVES

Conclusions

The conclusion of this thesis brings together a comprehensive exploration into the improvement of energy efficiency, sustainability, and occupant comfort within the realm of smart buildings. As introduced earlier, across multiple chapters, this research primarily addresses two significant challenges: the understanding of user interactions with the building, and the occupant privacy and comfort. And it also address two optimization problems: enhancing building energy efficiency and obtaining accurate and diversified data.

In Chapter 1, our journey began with an exhaustive literature review detailing energy management in smart buildings. Smart buildings are poised at the intersection of technology, sustainability, and human experience. They are not merely architectural marvels but have a pivotal role in global energy optimization. Recognizing and addressing challenges such as user-centric control, sensor placement, and multi-occupant activity detection remains paramount. A profound understanding of various communication systems and sensors, coupled with a detailed analysis of user behaviors, is central to this discourse. This chapter laid the groundwork, identifying key research gaps that guided our subsequent exploration, emphasizing that to drive the future of sustainable living, the buildings must evolve, considering both technological advances and the very humans they house.

Considering the current state of the art, optimal sensor placement and occupants activities detection have been identified as challenges to study in the thesis, Chapter 2 introduces two instrumental case studies which serve as the backbone of our analysis. Harnessing the power of DesignBuilder, we simulated energy consumption patterns for various occupancy scenarios. This approach allowed for a versatile examination of energy usage and management, which would later serve as training data for our machine learning occupants activities detection algorithms. Our real-world case, however, anchored us, ensuring our theoretical conclusions remained grounded, relevant, and applicable. The dynamic between these two methods — simulation and real - world observation — proved invaluable.

The dilemma of optimal sensor placement, as dissected in Chapter 3, is the crux of efficient energy management in smart environments. Ensuring the correct placement is not just a matter of technological prowess but requires an intricate understanding of environmental behavior and user interactions. It also directly link to the occupancy privacy and comfort with the question of data quality. Our exploration endorsed the Effective Independence Method and Information Entropy as standout solutions. Furthermore, we have given our definition for the word 'optimal', which is the information independence and stability. Our dive into complex scenarios like multi-wall environments validated the versatility of our chosen techniques. The conclusion from this chapter which the optimality of the locations chosen by our algorithm was verified through different experiments from simulation and real datasets.

In the age of artificial intelligence, the fourth chapter emphasized the irreplaceable role of machine learning and statistical methods in energy management. Through a meticulous process, we identified DBSCAN, PCA, MGD, Logistic Regression, ANN, XGBoost, and AutoEncoder as potential candidates for various prediction tasks. We benchmarked these algorithms to assess their strengths and weaknesses, ranging from their accuracy to adaptability. Given DBSCAN's superior detection accuracy and its unique capability to identify clusters of varied shapes, combined with the potential for significant speed-up via precomputed distance matrices, it emerges as a highly suitable choice. On the other hand, XGBoost consistently delivers commendable performance across diverse datasets. Its inherent design incorporates L1 and L2 regularization, offering a robust countermeasure against overfitting. Additionally, its parallel processing capabilities ensure swift computation, further enhancing its appeal. Consequently, we selected DBSCAN and XGBoost as interesting methods to detect occupants presence or window openings.

Based on selected algorithm, Chapter 5 represented a synthesis of our cumulative insights, culminating in the challenge of detecting multi-occupant behaviors. The emphasis on multi-occupant scenarios is rooted in the real-world complexities, be they office spaces, residential apartments, or educational institutions. Unlike single-occupant detection, multi-occupancy introduces intricacies that demand higher granularity in observation, interpretation, and response. At the same time, we have once again emphasized occupational privacy, so we need to use unsupervised or very little labeled data to detect multi-occupant behaviors. Our exploration began with the self-supervised learning approach, utilizing ANN XGBoost combined with time shuffling. In the context of occupant behavior, this approach offers the possibility of buildings 'learning' from the limited la-

beled data. The application of time shuffling was a particularly innovative touch, allowing the model to gain a deeper understanding of time-bound behaviors, which are central to energy use patterns. The self supervised showed excellent performance, which surpassing many traditional approaches even with only 5% labeled data to train the model. Then, we introduced the composite ANN-DBSCAN-HOTSPOT-BN method which combined the clustering capability of DBSCAN, the predictive prowess of ANNs, the root cause analysis of Hotspot, and the statistical depth of BN. Finally, ANN-DBSCAN-HOTSPOT-BN demonstrated a 45% accuracy in distinguishing different activities. Although the results from self-learning are much better, it is moving in the right direction as an unsupervised method to distinguish different activities.

In conclusion, this thesis represents a significant contribution to the field of smart buildings, developing and validating novel methodologies that solve the challenge for interaction between buildings and occupants, occupants privacy and optimal data. It paved the road for the user centric optimization control.

Perspectives

This thesis has several promising perspectives. In the Chapter 3, EIM using target sensors inside the room, in order to express the dynamic human behaviors, can be considered to develop dynamic sensor networks that can adjust their sensitivity or focus based on real-time data or feedback loops. This might involve sensors that can "communicate" with one another to optimize data capture based on observed movements or behaviors, and introducing analytics that predict human movement patterns might also be beneficial. By understanding typical trajectories or paths individuals take in a room, sensor placements can be preemptively optimized for expected behaviors. And one way to further enhance optimality is by using reinforcement learning techniques, where sensor placements and parameters are iterative fine-tuned based on real-time feedback loops. Because there is no underlying assumption for specific environment, EIM and information entropy methods are adaptable to other case studies, but given the different communication systems and sensor types discussed, when deploying the methods in a new building environment, it might be necessary to customize the choice of communication systems and sensors based on the building's size, or infrastructure.

In the Chapter 4 and 5, we expect the unsupervised and supervised methods selected (like DBSCAN or XGBoost) can be broadly applied to other scenarios. But, parameter

tuning and adjustments will be necessary based on the specifics of the new data and objectives. In the multi-activities task, trying to use unsupervised method or only few label to detection multi-activities is a challenge work, for the self learning supervised method. the idea can be broadly applied to other scenarios, but the performance is highly related to the pretext, so based on the specifics of different scenarios, finding an appropriate pretext is always worth trying. For the ANN-DBSCAN-HOTSPOT-BN method, within the complexity environment, it is not enough to determine root causes only through theory. One possible solution is using experiment by applying different activities in the room and combined with reinforcement learning to find the real root causes, which will greatly increase the detection capability.

With the detected user information, all the outputs should be seamlessly integrated with BMS, ensuring that insights drawn from data lead to actionable outcomes, such as adjusting HVAC, lighting, heating, or cooling systems. This would involve checking communication protocols, data formats, and integration capabilities. BMS is a system which integrated data network and control system for automation, monitoring, and control of HVAC, lighting, and other functions in a building. Control in building management is important for ensuring the effectiveness and efficiency of operations, reliable financial reporting, and compliance with laws, regulations, and policies. Based on the discussion and literature review in Chapter 1, the MPC is a feasible and reliable method in our application. One of the key advantages of MPC in the operation of HVAC systems is its ability to consider constraints, predict disturbances, and balance multiple conflicting objectives, such as indoor thermal comfort and building energy demand. In addition, MPC has been successfully implemented for building thermal regulation, maximizing the potential of building thermal mass (Serale et al.2018). An introduction to MPC can be found in Appendix C.1. Even though, various works have investigated the application of MPC algorithms in building control, it still facing some challenges, such as building modeling and integration of the user information. One suggestion approach for building modeling (in appendix C.2) and integration of the user information (in appendix C.3) can be found in Appendix.

The integration of user information into BMS via MPC represents a crucial step towards achieving energy-efficient and user-centric building operations. Firstly, an accurate building modeling is most important. The adoption of grey-box models, particularly the Resistance and Capacitance (RC) models, offers a promising balance between model accuracy and computational feasibility. By bridging the gap between data-driven black-box

models and physics-based white-box models, grey-box models ensure that the dynamic thermal behavior of buildings is captured with reasonable accuracy while remaining computationally efficient. Then, we need to estimate the model parameters, particularly using optimization techniques like Genetic Algorithm (GA), and further refines these models by fine-tuning model parameters based on observed building data.

With the increasing availability of user occupancy data, thanks to advancements in sensor technology and occupancy detection methods, the potential for optimizing building operations based on real-time user information has become more evident. Integrating this information into MPC allows for dynamic adjustment of building operational constraints based on actual and predicted occupancy, ensuring both energy efficiency and occupant comfort. Through the dynamic setting of temperature constraints, buildings can transition between comfort and setback modes, optimizing energy usage during unoccupied periods without compromising user comfort when the building is occupied.

However, the practical implementation of user-centric MPC in BMS is far from trivial. As highlighted, while the foundational concepts are established, real-world applications involve a myriad of interacting activities, with each potentially introducing additional constraints and considerations. For instance, the influence of different building zones, variability in user preferences, and external factors like weather conditions further complicate the decision-making process. As the research landscape continues to evolve, it is imperative to focus on developing holistic models that can seamlessly integrate diverse user information and adapt dynamically to ever-changing building environments. Furthermore, the advancement of efficient computational techniques and robust optimization algorithms will be crucial in ensuring the scalability and real-time applicability of such integrated systems.

In summary, the integration of user information into BMS using MPC holds immense promise for the future of sustainable and user-centric building operations. While challenges remain, as research continues in this direction, it is anticipated that the benefits of such integration, both in terms of energy savings and enhanced occupant comfort, will become increasingly evident in real-world applications.

Bibliography

- Abrishambaf, O. et al., « Application of a Home Energy Management System for Incentive-Based Demand Response Program Implementation », *in: 27th International Workshop on Database and Expert Systems Applications (DEXA)* (2016), pp. 153–157, DOI: 10.1109/DEXA.2016.043.
- Agency, European Environment, « Energy and climate change — European Environment Agency », *in: EEA Signals 2017* (2017), URL: <https://www.eea.europa.eu/signals/signals-2017/articles/energy-and-climate-change>.
- Agency, International Energy, *World Energy Outlook 2021*, IEA Publications, 2021, URL: <https://www.iea.org/reports/world-energy-outlook-2021>.
- Aghemo, C. et al., « Management and monitoring of public buildings through ICT based systems: Control rules for energy saving with lighting and HVAC services », *in: Frontiers of Architectural Research 2.2* (2013), pp. 147–161.
- Agrawal, Rakesh, Tomasz Imieliński, and Arun Swami, « Mining association rules between sets of items in large databases », *in: ACM SIGMOD Record 22.2* (1993), pp. 207–216.
- Ahn, K.U. and C.-S. Park, « Temporal and spatial variation in the predictability of building occupancy », *in: Building and Environment 149* (2018), pp. 477–489, DOI: 10.1016/j.buildenv.2018.12.043.
- Alam, M.R., M.B.I. Reaz, and M.A.M. Ali, « A review of smart homes—Past, present, and future », *in: IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42.6* (2012), pp. 1190–1203, DOI: 10.1109/TSMCC.2012.2189204.
- Alhamoud, A. et al., « SMARTENERGY.KOM: An intelligent system for energy saving in smart home », *in: 39th Annual IEEE Conference on Local Computer Networks Workshops*, 2014, pp. 685–692, DOI: 10.1109/LCNW.2014.6927721.
- Aliero, M.S. et al., « Non-Intrusive Room Occupancy Prediction Performance Analysis Using Different Machine Learning Techniques », *in: Energies 15.9* (2022), p. 9231, DOI: 10.3390/en15239231.
- Alsalemi, A. et al., « Endorsing domestic energy saving behavior using micro-moment classification », *in: Applied Energy 250* (2019), pp. 1302–1311, DOI: 10.1016/j.apenergy.2019.05.089.
- Alsalemi, A. et al., « Smart energy usage and visualization based on micro-moments », *in: Intelligent Systems and Applications: Proceedings of the 2019 Intelligent Systems*

-
- Conference (IntelliSys)*, vol. 2, 2020, pp. 557–566, DOI: 10.1007/978-3-030-29513-4_41.
- Amayri, M. et al., « Bayesian network and Hidden Markov Model for estimating occupancy from measurements and knowledge », in: *2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2017, pp. 690–695, DOI: 10.1109/IDAACS.2017.8095179.
- Amayri, M. et al., « Estimating Occupancy from Measurements and Knowledge Using the Bayesian Network for Energy Management », in: *Journal of Sensors* 7129872 (2019), p. 12, DOI: 10.1155/2019/7129872.
- Amayri, Manar et al., « Estimating occupancy from measurements and knowledge using the bayesian network for energy management », in: *Journal of Sensors* 2019 (2019).
- American Society of Heating, Refrigerating and Air-Conditioning Engineers, *2020 ASHRAE Handbook: HVAC Systems and Equipment*, Ashrae, 2020.
- Amini, Amineh, Teh Ying Wah, and Hadi Saboohi, « On Density-Based Data Streams Clustering Algorithms: A Survey », in: *Journal of Computer Science and Technology* 29.1 (2014), pp. 116–141, DOI: 10.1007/s11390-013-1416-3.
- Arendt, K. et al., « ModestPy: An Open-Source Python Tool for Parameter Estimation in Functional Mock-up Units », in: *Proceedings of the American Modeling Conference 2018*, vol. 154, Somb. Conference Center, Cambridge MA, USA, 2019, pp. 121–30, DOI: 10.3384/ecp18154121.
- Arghira, N. et al., « Prediction of appliances energy use in smart homes », in: *Energy* 48.1 (2012), pp. 128–134, DOI: 10.1016/j.energy.2012.04.010.
- Arief-Ang, I.B., M. Hamilton, and F.D. Salim, « RUP: Large Room Utilisation Prediction with carbon dioxide sensor », in: *Pervasive and Mobile Computing* 46 (2018), pp. 49–72, DOI: 10.1016/j.pmcj.2018.03.001.
- Arief-Ang, I.B., F.D. Salim, and M. Hamilton, « DA-HOC: semi-supervised domain adaptation for room occupancy prediction using CO2 sensor data », in: *Proceedings of the 4th ACM International Conference on Systems for Energy-Efficient Built Environments (BuildSys '17)*, 2017, pp. 1–10, DOI: 10.1145/3137133.3137146.
- Arman, Utami Dewi, Jihan Melasari, and Aldan Roby Suwanda, « Identifikasi Penyebab Kecelakaan Kerja Konstruksi Menggunakan Accident Root Cause Tracing Model (ARCTM) dan Fault Tree Analysis (FTA) », in: *Cantilever* 11.1 (2022), DOI: 10.35139/cantilever.v11i1.112.

-
- ASHRAE, « ASHRAE Handbook », *in: ASHRAE* (2020), URL: <https://www.ashrae.org/technical-resources/bookstore/2020-ashrae-handbook-hvac-systems-and-equipment>.
- « Thermal Environmental Conditions for Human Occupancy », *in: ASHRAE* (2017), URL: <https://www.ashrae.org/technical-resources/bookstore/standard-55-thermal-environmental-conditions-for-human-occupancy>.
- *Thermal Environmental Conditions for Human Occupancy*, 2017.
- Bae, Y. et al., « Sensor impacts on building and HVAC controls: A critical review for building energy performance », *in: Advances in Applied Energy* 4 (2021), p. 100068, DOI: 10.1016/j.adapen.2021.100068.
- Bae, Yeonjin et al., « Sensor impacts on building and HVAC controls: A critical review for building energy performance », *in: Advances in Applied Energy* 4 (2021), p. 100068, DOI: 10.1016/j.adapen.2021.100068.
- Baird, Z. et al., « Principal component analysis-based occupancy detection with ultra wideband radar », *in: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2017, pp. 1573–1576, DOI: 10.1109/MWSCAS.2017.8053237.
- Balta-Ozkan, N., O. Amerighi, and B. Boteler, « A comparison of consumer perceptions towards smart homes in the UK, Germany and Italy: reflections for policy and future research », *in: Technology Analysis and Strategic Management* 26.10 (2014), DOI: 10.1080/09537325.2014.975788.
- Bao, R. and Z. Yang, « CNN-Based Regional People Counting Algorithm Exploiting Multi-Scale Range-Time Maps With an IR-UWB Radar », *in: IEEE Sensors Journal* 21.12 (2021), pp. 13704–13713, DOI: 10.1109/JSEN.2021.3071941.
- Barbato, A. et al., « Home energy saving through a user profiling system based on wireless sensors », *in: Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings (BuildSys '09)*, 2009, pp. 49–54, DOI: 10.1145/1810279.1810291.
- Batra, N. et al., « A comparison of non-intrusive load monitoring methods for commercial and residential buildings », *in: arXiv preprint arXiv:1408.6595* (2014).
- Bavaresco, M.V. et al., « Technological innovations to assess and include the human dimension in the building-performance loop: A review », *in: Energy and Buildings* 202 (2019), DOI: 10.1016/j.enbuild.2019.109365.

-
- Bazazzadeh, H. et al., « The Impact Assessment of Climate Change on Building Energy Consumption in Poland », *in: Energies* 14.14 (2021), p. 4084, DOI: 10.3390/EN14144084.
- Bazazzadeh, Hassan, Adam Nadolny, and Seyedeh Sara Hashemi Safaei, « Climate Change and Building Energy Consumption: A Review of the Impact of Weather Parameters Influenced by Climate Change on Household Heating and Cooling Demands of Buildings », *in: European Journal of Sustainable Development* 10.2 (2021), pp. 1–12, DOI: 10.14207/ejsd.2021.v10n2p1.
- Bhagwan, Ranjita et al., « Adtributor: Revenue Debugging in Advertising Systems », *in: 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, 2014, pp. 43–55.
- Bianco, Simone and Francesco Tisato, « Sensor placement optimization in buildings », *in: Proceedings of SPIE - The International Society for Optical Engineering*, vol. 8300, 2012, pp. 1–, DOI: 10.1117/12.911021.
- Bigaud, D. et al., « Detection of Faults and Drifts in the Energy Performance of a Building Using Bayesian Networks », *in: ASME Journal of Dynamic Systems, Measurement, and Control* 141.10 (2019), p. 101011, DOI: 10.1115/1.4043922.
- Biyik, Emrah and Aysegul Kahraman, « A predictive control strategy for optimal management of peak load, thermal comfort, energy storage and renewables in multi-zone buildings », *in: Journal of Building Engineering* 25 (2019), p. 100826, DOI: 10.1016/j.jobe.2019.100826.
- Blum, D.H. et al., « Practical factors of envelope model setup and their effects on the performance of model predictive control for building heating, ventilating, and air conditioning systems », *in: Appl Energy* 236 (2019), pp. 410–25, DOI: 10.1016/j.apenergy.2018.11.093.
- Boniol, Paul et al., « Unsupervised and scalable subsequence anomaly detection in large data series », *in: The VLDB Journal* (2021), pp. 1–23.
- Bortolini, R. and N. Forcada, « A probabilistic-based approach to support the comfort performance assessment of existing buildings », *in: Journal of Cleaner Production* 237 (2019), p. 117720, DOI: 10.1016/j.jclepro.2019.117720.
- Brooks, J. et al., « Energy-efficient control of under-actuated HVAC zones in commercial buildings », *in: Energy and Buildings* 93 (2015), pp. 160–168, DOI: 10.1016/j.enbuild.2015.01.050.

-
- Browne, C.B. et al., « A Survey of Monte Carlo Tree Search Methods », *in: IEEE Transactions on Computational Intelligence and AI in Games* 4.1 (2012), pp. 1–43, DOI: 10.1109/TCIAIG.2012.2186810.
- Brun, A. et al., « Summer Comfort in a Low-Inertia Building with a New Free-Cooling System », *in: Applied Energy* 112 (2013), pp. 338–349, DOI: 10.1016/j.apenergy.2013.05.052.
- Burhanuddin, A. and W. Sulistiyowati, « Quality Control Design to Reduce Shoes Production Defects Using Root Cause Analysis and Lean Six Sigma Methods », *in: PELS* 2.2 (2022), DOI: 10.21070/pels.v2i2.1242.
- Candanedo, L.M. and V. Feldheim, « Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models », *in: Energy and Buildings* 112 (2016), pp. 28–39, DOI: 10.1016/j.enbuild.2015.11.071.
- Candanedo, L.M., V. Feldheim, and D. Deramaix, « A methodology based on Hidden Markov Models for occupancy detection and a case study in a low energy residential building », *in: Energy and Buildings* 148 (2017), pp. 327–341, DOI: 10.1016/j.enbuild.2017.05.031.
- « Data driven prediction models of energy use of appliances in a low-energy house », *in: Energy and Buildings* 140 (2017), pp. 81–97, DOI: 10.1016/j.enbuild.2017.01.083.
- Cao, L. et al., « Electrical load prediction of healthcare buildings through single and ensemble learning », *in: Energy Reports* 6 (2020), pp. 2751–2767, DOI: 10.1016/j.egy.2020.10.005.
- Castello, Charles C et al., « Optimal sensor placement strategy for environmental monitoring using wireless sensor networks », *in: 2010 42nd Southeastern Symposium on System Theory (SSST)*, IEEE, 2010, pp. 275–279.
- Castro-Triguero, R. et al., « Robustness of optimal sensor placement under parametric uncertainty », *in: Mechanical Systems and Signal Processing* 41 (2013), pp. 268–287, DOI: 10.1016/j.ymsp.2013.06.022.
- Chai, W. et al., « Optimal sensor placement of bridge structure based on sensitivity-effective independence method », *in: IET Circuits Devices Systems* 16 (2021), pp. 125–135, DOI: 10.1049/cds2.12078.
- Chalapathy, R., N.L.D. Khoa, and S. Sethuvenkatraman, « Comparing multi-step ahead building cooling load prediction using shallow machine learning and deep learning

-
- models », *in: Sustainable Energy, Grids and Networks* 28 (2021), p. 100543, DOI: 10.1016/j.segan.2021.100543.
- Chandra, Rohitash, Shaurya Goyal, and Rishabh Gupta, « Evaluation of Deep Learning Models for Multi-Step Ahead Time Series Prediction », *in: IEEE Access*, 2021, DOI: 10.1109/ACCESS.2021.3085085.
- Chandran, A.K. et al., « A PTZ Camera Based People-Occupancy Estimation System (PCBPOES) », *in: 15th IAPR International Conference on Machine Vision Applications (MVA)*, Nagoya, Japan, 2017, 4 pages.
- Chawla, N.V. et al., « SMOTE: synthetic minority over-sampling technique », *in: Journal of artificial intelligence research* 16 (2002), pp. 321–357, DOI: 10.1613/jair.953.
- Che, Zhengping et al., « Recurrent Neural Networks for Multivariate Time Series with Missing Values », *in: Scientific Reports* 8.1 (2018), p. 6085, DOI: 10.1038/s41598-018-24271-9.
- Chen, M.T. and C.M. Lin, « Standby Power Management of a Smart Home Appliance by Using Energy Saving System With Active Loading Feature Identification », *in: IEEE Transactions on Consumer Electronics* 65.1 (2018), pp. 11–17, DOI: 10.1109/TCE.2018.2885034.
- Chen, S. et al., « Prediction of office building electricity demand using artificial neural network by splitting the time horizon for different occupancy rates », *in: Energy and AI* 5 (2021), p. 100093, DOI: 10.1016/j.egyai.2021.100093.
- Chen, Ting et al., « A Simple Framework for Contrastive Learning of Visual Representations », *in: arXiv preprint arXiv:2002.05709* (2020).
- Chen, X. and X. Li, « Virtual temperature measurement for smart buildings via Bayesian model fusion », *in: 2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, pp. 950–953.
- Chen, Z. and C. Jiang, « Building Occupancy Modeling Using Generative Adversarial Network », *in: Energy and Buildings* 174 (2018), pp. 372–379, DOI: 10.1016/j.enbuild.2018.06.029.
- Chen, Z., C. Jiang, and L. Xie, « Building occupancy estimation and detection: a review », *in: Energy and Buildings* 169 (2018), pp. 260–270, DOI: 10.1016/j.enbuild.2018.03.084.
- Chen, Z., M.K. Masood, and Y.C. Soh, « A fusion framework for occupancy estimation in office buildings based on environmental sensor data », *in: Energy and Buildings* 133 (2016), pp. 790–798, DOI: 10.1016/j.enbuild.2016.10.030.

-
- Chen, Z. and Y.C. Soh, « Comparing occupancy models and data mining approaches for regular occupancy prediction in commercial buildings », *in: Journal of Building Performance Simulation* 10.5-6 (2017), pp. 545–553, DOI: 10.1080/19401493.2016.1199735.
- Cho, Brian et al., « Effective Missing Value Imputation Methods for Building Monitoring Data », *in: 2020 IEEE International Conference on Big Data (Big Data)*, 2020, pp. 2866–2875, DOI: 10.1109/BigData50022.2020.9378230.
- Cho, S. et al., « Wireless, AI-enabled wearable thermal comfort sensor for energy-efficient, human-in-the-loop control of indoor temperature », *in: Biosensors and Bioelectronics* 223 (2023), p. 115018, DOI: 10.1016/j.bios.2022.115018.
- Chokwitthaya, C. et al., « Improving Prediction Accuracy in Building Performance Models Using Generative Adversarial Networks (GANs) », *in: arXiv preprint arXiv:1906.05767v2* (2019).
- Chong, Adrian et al., « Imputation of Missing Values in Building Sensor Data », *in: IBPSA-USA SimBuild 2016*, Salt Lake City, 2016, p. 8.
- Cody, Chand et al., « Privacy Preserving Occupancy Detection Using NB IoT Sensors », *in: IEEE Canadian Conference of Electrical and Computer Engineering*, Virtual (Online), Sept. 2021, DOI: 10.1109/CCECE53047.2021.9569139.
- Commission, European, « Guidelines for the transposition of the new Energy Performance Buildings Directive (EU) 2018/844 in Member States », *in: European Commission Reports* (2019), URL: https://ec.europa.eu/energy/sites/default/files/documents/1_en_act_part1_v3.pdf.
- Conti, F., A. Pullini, and L. Benini, « Brain-Inspired Classroom Occupancy Monitoring on a Low-Power Mobile Platform », *in: 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Columbus, OH, USA, 2014, pp. 624–629, DOI: 10.1109/CVPRW.2014.95.
- Cottone, P. et al., « User activity recognition for energy saving in smart homes », *in: 2013 Sustainable Internet and ICT for Sustainability (SustainIT)*, Palermo, Italy, 2013, pp. 1–9, DOI: 10.1109/SustainIT.2013.6685196.
- D’Oca, S. and T. Hong, « Occupancy schedules learning process through a data mining framework », *in: Energy and Buildings* 88 (2015), pp. 395–408, DOI: 10.1016/j.enbuild.2014.11.065.

-
- Dahbi, Azzeddine et al., « Finding Suitable Threshold for Support in Apriori Algorithm Using Statistical Measures », *in: Algorithms for Intelligent Systems*, Springer, 2021, pp. 85–95.
- Dai, X., J. Liu, and X. Zhang, « A review of studies applying machine learning models to predict occupancy and window-opening behaviours in smart buildings », *in: Energy and Buildings* 223 (2020), p. 110159, DOI: 10.1016/j.enbuild.2020.110159.
- Danaei-Mehr, H., H. Polat, and A. Cetin, « Resident activity recognition in smart homes by using artificial neural networks », *in: 2016 4th International Istanbul Smart Grid Congress and Fair (ICSG)*, Istanbul, Turkey, 2016, pp. 1–5, DOI: 10.1109/SGCF.2016.7492428.
- Das, S.K., N. Roy, and A. Roy, « Context-aware resource management in multi-inhabitant smart homes: A framework based on Nash H-learning », *in: Pervasive and Mobile Computing* 2.4 (2006), pp. 372–404, DOI: 10.1016/j.pmcj.2006.08.003.
- De Coninck, R. and L. Helsen, « Practical implementation and evaluation of model predictive control for an office building in Brussels », *in: Energy and Buildings* 111 (2016), pp. 290–298, DOI: 10.1016/j.enbuild.2015.11.014.
- Delaney, D.T., G.M.P. O’Hare, and A.G. Ruzzelli, « Evaluation of energy-efficiency in lighting systems using sensor networks », *in: Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, 2009, pp. 61–66.
- Delzendeh, Elham et al., « The impact of occupants’ behaviours on building energy analysis: A research review », *in: Renewable and Sustainable Energy Reviews* 80 (2017), pp. 1061–1071, DOI: 10.1016/j.rser.2017.05.264.
- Devlin, Jacob et al., « BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding », *in: NAACL-HLT*, 2019.
- Diraco, G., A. Leone, and P. Siciliano, « People occupancy detection and profiling with 3D depth sensors for building energy management », *in: Energy and Buildings* 92 (2015), pp. 246–266, DOI: 10.1016/j.enbuild.2015.01.043.
- Diyan, M., B.N. Silva, and K. Han, « A Multi-Objective Approach for Optimal Energy Management in Smart Home Using the Reinforcement Learning », *in: Sensors (Basel)* 20.12 (2020), p. 3450, DOI: 10.3390/s20123450.
- Doebeling, Scott W, Lee D Peterson, and Kenneth F Alvin, « Estimation of reciprocal residual flexibility from experimental modal data », *in: AIAA journal* 34.8 (1996), pp. 1678–1685.

-
- Doersch, Carl, Abhinav Gupta, and Alexei A. Efros, « Unsupervised Visual Representation Learning by Context Prediction », *in: ICCV 2015*, 2015.
- Dong, B. et al., « A Global Building Occupant Behavior Database », *in: Scientific Data* 9.1 (2022), p. 369, DOI: 10.1038/s41597-022-01475-3.
- Dong, XY et al., « Distance coefficient-Fisher information criterion for optimal sensor placement », *in: CAAI Trans. Intell. Syst* 12 (2017), pp. 32–37.
- Dorokhova, M., C. Ballif, and N. Wyrsh, « Rule-based scheduling of air conditioning using occupancy forecasting », *in: Energy and AI* 2 (2020), p. 100022, DOI: 10.1016/j.egyai.2020.100022.
- Dounis, A.I. et al., « Intelligent control system for reconciliation of the energy savings with comfort in buildings using soft computing techniques », *in: Energy and Buildings* 43.1 (2011), pp. 66–74, DOI: 10.1016/j.enbuild.2010.08.014.
- Drgoňa, Ján et al., « All you need to know about model predictive control for buildings », *in: Annual Reviews in Control* 50 (2020), pp. 142–154, DOI: 10.1016/j.arcontrol.2020.09.001.
- Dridi, J., M. Amayri, and N. Bouguila, « Transfer learning for estimating occupancy and recognizing activities in smart buildings », *in: Building and Environment* 217 (2022), p. 109057, DOI: 10.1016/j.buildenv.2022.109057.
- Du, Z. et al., « Temperature sensor placement optimization for VAV control using CFD–BES co-simulation strategy », *in: Building and Environment* 85 (2015), pp. 104–113, DOI: 10.1016/j.buildenv.2014.11.033.
- Edwards, David, John Mottershead, and Michael Friswell, « Optimal sensor placement for structural health monitoring using genetic algorithms », *in: Structural Health Monitoring* 16.6 (2017), pp. 677–692.
- Eguaras-Martínez, M., M. Vidaurre-Arbizu, and C. Martín-Gómez, « Simulation and evaluation of Building Information Modeling in a real pilot site », *in: Applied Energy* 114.C (2014), pp. 475–484, DOI: 10.1016/j.apenergy.2013.09.047.
- Erickson, V.L. et al., « Energy efficient building environment control strategies using real-time occupancy measurements », *in: Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, 2009, pp. 19–24, DOI: 10.1145/1810279.1810284.
- Ester, Martin et al., « A density-based algorithm for discovering clusters in large spatial databases with noise », *in: kdd*, vol. 96, 34, 1996, pp. 226–231.

-
- Fabi, Valentina, Rune Vinther Andersen, and Stefano Paolo Corgnati, « Main physical environmental variables driving occupant behaviour with regard to natural ventilation », *in: Energy and Buildings* 67 (2013), pp. 166–176, DOI: 10.1016/j.enbuild.2013.08.017.
- Fairey, Philip and David Goldstein, *Metrics for Energy Efficient Buildings: How Do We Measure Efficiency?*, tech. rep., Florida Solar Energy Center, 2016.
- Fajilla, G. et al., « Assessment of probabilistic models to estimate the occupancy state in office buildings using indoor parameters and user-related variables », *in: Energy and Buildings* 246 (2021), p. 111105, DOI: 10.1016/j.enbuild.2021.111105.
- Fakhar, M.Z., E. Yalcin, and A. Bilge, « A survey of smart home energy conservation techniques », *in: Expert Systems with Applications* 213.B (2023), p. 118974, DOI: 10.1016/j.eswa.2022.118974.
- Fang, H. and C. Hu, « Recognizing human activity in smart home using deep learning algorithm », *in: Proceedings of the 33rd Chinese Control Conference*, Nanjing, China, 2014, pp. 4716–4720, DOI: 10.1109/ChiCC.2014.6895735.
- Fang, T. and R. Lahdelma, « Evaluation of a multiple linear regression model and SARIMA model in forecasting heat demand for district heating system », *in: Applied Energy* 179 (2016), pp. 544–552, DOI: 10.1016/j.apenergy.2016.06.133.
- Faridah, D. et al., « Optimal thermal sensors placement based on indoor thermal environment characterization by using CFD model », *in: Journal of Applied Engineering Science* 19.3 (2021), pp. 628–641, DOI: 10.5937/jaes0-28985.
- Fisher, Ronald A., « On the Mathematical Foundations of Theoretical Statistics », *in: Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 222 (1922), pp. 309–368.
- Fouladgar, Nazanin and Kary Främling, « A Novel LSTM for Multivariate Time Series with Massive Missingness », *in: Sensors* 20.10 (2020), p. 2832, DOI: 10.3390/s20102832.
- Franco, A. and F. Leccese, « Measurement of CO₂ concentration for occupancy estimation in educational buildings with energy efficiency purposes », *in: Journal of Building Engineering* 32 (2020), p. 101714, DOI: 10.1016/j.jobe.2020.101714.
- Frei, M. et al., « Sustainable Buildings: Opportunities and Challenges for New Buildings and Retrofits », *in: Frontiers in Built Environment* 6 (2021), DOI: 10.3389/fbuil.2020.609877.

-
- Freund, S. and G. Schmitz, « Development of a Framework for Model Predictive Control (MPC) in a Large-Sized Low-Energy Office Building Using Modelica Grey-Box Models », *in: 16th Conference of IBPSA*, IBPSA, Rome, Italy, 2019, pp. 2864–71.
- Gade, R., A. Jørgensen, and T.B. Moeslund, « Occupancy Analysis of Sports Arenas Using Thermal Imaging », *in: Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*, Roma, Italy, 2012, pp. 277–283.
- Gan, Lu et al., « Balancing of supply and demand of renewable energy power system: review and bibliometric analysis », *in: Sustainable Futures 1.1* (2020), p. 100002.
- Gander, Walter, « Algorithms for the QR decomposition », *in: Res. Rep 80.02* (1980), pp. 1251–1268.
- Gautam, D.K., P. Kotecha, and S. Subbiah, « Efficient k-means clustering and greedy selection-based reduction of nodal search space for optimization of sensor placement in the water distribution networks », *in: Water Research 220* (2022), p. 118666, DOI: 10.1016/j.watres.2022.118666.
- Ghaffar, M. et al., « Non-Intrusive Load Monitoring of Buildings Using Spectral Clustering », *in: Sensors 22* (2022), p. 4036, DOI: 10.3390/s22114036.
- Gidaris, Spyros et al., « Boosting Few-Shot Visual Learning with Self-Supervision », *in: ICCV 2019*, 2019.
- « Unsupervised Representation Learning by Predicting Image Rotations », *in: ICLR 2018*, 2018.
- Goldstein, Rafael, Alexandre Tessier, and Ahmed Khan, « Space layout in occupant behavior simulation », *in: Proceedings of the 12th Conference of International Building Performance Simulation Association*, IBPSA, 2011, pp. 1073–1080.
- Gruber, M., A. Trüschel, and J.-O. Dalenbäck, « CO2 sensors for occupancy estimations: Potential in building automation applications », *in: Energy and Buildings 84* (2014), pp. 548–556, DOI: 10.1016/j.enbuild.2014.09.002.
- Guhr, N. et al., « Privacy concerns in the smart home context », *in: SN Applied Sciences 2.2* (2020), p. 247, DOI: 10.1007/s42452-020-2025-8.
- Guo, X. et al., « The performance of occupancy-based lighting control systems: a review », *in: Lighting Research Technology 42.4* (2010), pp. 415–431, DOI: 10.1177/1477153510376225.
- Guyan, Robert J, « Reduction of stiffness and mass matrices », *in: AIAA journal 3.2* (1965), pp. 380–380.

-
- Gyalistras, D. et al., « Analysis of energy savings potentials for integrated room automation », *in: Clima - RHEVA World Congress*, Antalya, Turkey, 2010.
- Hafeez, G. et al., « Efficient Energy Management of IoT-Enabled Smart Homes Under Price-Based Demand Response Program in Smart Grid », *in: Sensors* 20.11 (2020), p. 3155, DOI: 10.3390/s20113155.
- Hammoud, A., M. Deriaz, and D. Konstantas, « UltraSense: A Self-Calibrating Ultrasound-Based Room Occupancy Sensing System », *in: Procedia Computer Science* 109 (2017), pp. 75–83, DOI: 10.1016/j.procs.2017.05.297.
- Harb, H. et al., « Development and validation of grey-box models for forecasting the thermal response of occupied buildings », *in: Energy Build* 117 (2016), pp. 199–207, DOI: 10.1016/j.enbuild.2016.02.021.
- Hasan, Md. Kamrul et al., « Missing Value Imputation Affects the Performance of Machine Learning: A Review and Analysis of the Literature (2010–2021) », *in: Informatics in Medicine Unlocked* 27 (2021), p. 100799, DOI: 10.1016/j.imu.2021.100799.
- Hassani, S. and U. Dackermann, « A Systematic Review of Optimization Algorithms for Structural Health Monitoring and Optimal Sensor Placement », *in: Sensors* 23.6 (2023), p. 3293, DOI: 10.3390/s23063293.
- Hayati, A. et al., « Sustainable energy management practices and its effect on EEI: a case on university buildings », *in: Journal of Modern Science and Technology* 2.1 (2014), pp. 39–48, ISSN: 2201-6686.
- He, Kaiming et al., « Momentum Contrast for Unsupervised Visual Representation Learning », *in: ArXiv* abs/1911.05722 (2019).
- He, L. et al., « Optimal multiaxial sensor placement for modal identification of large structures », *in: Structural Control and Health Monitoring* 21 (2014), pp. 61–79, DOI: 10.1002/stc.1550.
- Hedegaard, R.E. and S. Petersen, « Evaluation of Grey-Box Model Parameter Estimates Intended for Thermal Characterization of Buildings », *in: Energy Procedia* 132 (2017), pp. 982–7, DOI: 10.1016/j.egypro.2017.09.692.
- Hedegaard, R.E. et al., « Bottom-up modelling methodology for urban-scale analysis of residential space heating demand response », *in: Appl Energy* 242 (2019), pp. 181–204, DOI: 10.1016/j.apenergy.2019.03.063.
- Heo, G, ML Wang, and D Satpathi, « Optimal transducer placement for health monitoring of long span bridge », *in: Soil dynamics and earthquake engineering* 16.7-8 (1997), pp. 495–502.

-
- Heredia-Zavoni, E., R. Montes-Iturrizaga, and L. Esteva, « Optimal instrumentation of structures on flexible base for system identification », *in: Earthquake Engineering Structural Dynamics* 28.12 (1999), pp. 1471–1482, DOI: 10.1002/(SICI)1096-9845(199912)28:12<1471::AID-EQE872>3.0.CO;2-M.
- Himeur, Y. et al., « A Novel Approach for Detecting Anomalous Energy Consumption Based on Micro-Moments and Deep Neural Networks », *in: Cognitive Computation* 12 (2020), pp. 1381–1401, DOI: 10.1007/s12559-020-09764-y.
- Hjelm, R. Devon et al., « Learning deep representations by mutual information estimation and maximization », *in: ICLR 2019*, 2019.
- Holland, John H, « Genetic Algorithms », *in: Scientific American* 267.1 (1992), pp. 66–72.
- Hong, T. et al., « An ontology to represent energy-related occupant behavior in buildings. Part I: Introduction to the DNAs framework », *in: Building and Environment* 92 (2015), pp. 764–777, DOI: 10.1016/j.buildenv.2015.02.019.
- Hong, T. et al., « An ontology to represent energy-related occupant behavior in buildings. Part II: Implementation of the DNAs framework using an XML schema », *in: Building and Environment* 94.1 (2015), pp. 196–205, DOI: 10.1016/j.buildenv.2015.08.006.
- Hosamo, H.H. et al., « Improving building occupant comfort through a digital twin approach: A Bayesian network model and predictive maintenance method », *in: Energy and Buildings* 288 (2023), p. 112992, DOI: 10.1016/j.enbuild.2023.112992.
- Hou, R. et al., « Genetic algorithm based optimal sensor placement for L1-regularized damage detection », *in: Structural Control and Health Monitoring* 26.1 (2018), DOI: 10.1002/stc.2274.
- Hsien-te, Lin and Yen Chia-ju, « Hotel energy rating system using dynamic zone EUI method in Taiwan », *in: Energy and Buildings* 244 (2021), p. 111023.
- Hu, Maomao et al., « Price-responsive model predictive control of floor heating systems for demand response using building thermal mass », *in: Applied Thermal Engineering* 153 (2019), pp. 316–329, DOI: 10.1016/j.applthermaleng.2019.02.107.
- Huang, G., P. Zhou, and L. Zhang, « Optimal Location of Wireless Temperature Sensor Nodes in Large-scale Rooms », *in: Conference: Indoor Air* (2014), DOI: 10.13140/RG.2.1.1897.2884.
- Huang, G.-B., Q.-Y. Zhu, and C.-K. Siew, « Extreme learning machine: Theory and applications », *in: Neurocomputing* 70.1-3 (2006), pp. 489–501, DOI: 10.1016/j.neucom.2005.12.126.

-
- Huang, Qian, « Occupancy-driven energy-efficient buildings using audio processing with background sound cancellation », *in: Buildings* 8.6 (2018), p. 78.
- Huang, Sen, Wangda Zuo, and Michael D Sohn, « A Bayesian Network model for predicting cooling load of commercial buildings », *in: Building simulation*, vol. 11, Springer, 2018, pp. 87–101.
- Huchuk, B., S. Sanner, and W. O'Brien, « Comparison of machine learning models for occupancy prediction in residential buildings using connected thermostat data », *in: Building and Environment* 160 (2019), p. 106177, DOI: 10.1016/j.buildenv.2019.106177.
- Iannizzotto, G., L. Lo Bello, and A. Nucita, « Improving BLE-Based Passive Human Sensing with Deep Learning », *in: Sensors* 23 (2023), p. 2581, DOI: 10.3390/s23052581.
- IEA, *Building Energy Performance Metrics - Supporting Energy Efficiency Progress in Major Economies*, tech. rep., IEA, 2015, p. 110.
- *Digitalisation and Energy*, tech. rep., License: CC BY 4.0, IEA, 2017, p. 185, URL: <https://www.iea.org/reports/digitalisation-and-energy>.
- « Global Energy Review 2020 », *in: International Energy Agency* (2020), URL: <https://www.iea.org/reports/global-energy-review-2020>.
- IPCC, « Climate Change 2022: Impacts, Adaptation and Vulnerability. Working Group II Contribution to the IPCC Sixth Assessment Report », *in: IPCC* (2022), URL: <https://www.ipcc.ch/report/ar6/wg2/>.
- « Climate Change 2023: Synthesis Report. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change [Core Writing Team, H. Lee and J. Romero (eds.)] », *in: IPCC* (2023), URL: <https://www.ipcc.ch/report/ar6/syr/>.
- Islam, S.M.M. et al., « Building occupancy estimation using microwave Doppler radar and wavelet transform », *in: Building and Environment* 236 (2023), p. 110233, DOI: 10.1016/j.buildenv.2023.110233.
- ISO, « Lighting of indoor work places; ISO Standard 8995:2002 », *in: ISO* (2002), URL: <https://www.iso.org/standard/33333.html>.
- Jahn, M. et al., « The Energy Aware Smart Home », *in: 2010 5th International Conference on Future Information Technology*, 2010, pp. 1–8, DOI: 10.1109/FUTURETECH.2010.5482712.

-
- Javaid, N. et al., « Towards Cost and Comfort Based Hybrid Optimization for Residential Load Scheduling in a Smart Grid », *in: Energies* 10.10 (2017), p. 1546, DOI: 10.3390/en10101546.
- Jeyapadmini, J. and K.R. Kashwan, « Effective power utilization and conservation in smart homes using IoT », *in: 2015 International Conference on Computation of Power, Energy, Information and Communication (ICCPEIC)*, 2015, pp. 195–199, DOI: 10.1109/ICCPEIC.2015.7259463.
- Jin, Xin et al., « Foresee: A user-centric home energy management system for energy efficiency and demand response », *in: Applied Energy* 205 (2017), pp. 1583–1595, DOI: 10.1016/j.apenergy.2017.08.166.
- Judkoff, R. and J. Neymark, *International Energy Agency Building Energy Simulation-Test (BESTEST) and Diagnostic Method*, tech. rep. NREL/TP-472-6231, National Renewable Energy Laboratory, Golden, CO, 1995, URL: <http://www.nrel.gov/docs/legosti/old/6231.pdf>.
- « Twenty years on!: Updating the IEA BESTEST building thermal fabric test cases for ASHRAE standard 140 », *in: Proceedings of BS 2013: 13th Conference of the International Building Performance Simulation Association*, 2013, pp. 63–70.
- Juricic, Sarah et al., « Design of a Short Perturbation Method for On-Site Estimation of a Building Envelope Thermal Performance », *in: Energy and Buildings* 269 (2022), p. 112211, DOI: 10.1016/j.enbuild.2022.112211.
- Kalluri, B., « Optimal Sensor Placement Strategy for Office Buildings Using Clustering Algorithms », *in: Energy and Buildings* 158 (2017), pp. 1206–1225, DOI: 10.1016/j.enbuild.2017.10.074.
- Kammer, D.C., « Sensor placement for on-orbit modal identification and correlation of large space structures », *in: Journal of Guidance, Control, and Dynamics* 14.2 (1991), pp. 251–259, DOI: 10.2514/3.20635.
- Kampeidou, S.I. et al., « Real-time occupancy detection with physics-informed pattern-recognition machines based on limited CO₂ and temperature sensors », *in: Energy and Buildings* 242 (2021), p. 110863, DOI: 10.1016/j.enbuild.2021.110863.
- Kawakami, Tomoya et al., « A rule-based Home Energy Management System using the Rete algorithm », *in: IEEE Global Conference on Consumer Electronics (GCCE) 2* (2013), p. 6664785.

-
- Kennedy, James and Russell Eberhart, « Particle swarm optimization », *in: Proceedings of ICNN'95 - International Conference on Neural Networks*, IEEE, 1995, 1942–1948 vol.4, DOI: 10.1109/ICNN.1995.488968.
- Khan, W., S.S.W. Walker, and W. Zeiler, « A bottom-up framework for analysing city-scale energy data using high dimension reduction techniques », *in: Sustainable Cities and Society* 89 (2023), p. 104323, DOI: 10.1016/j.scs.2022.104323.
- Kim, J. et al., « Estimation of Occupancy Using IoT Sensors and a Carbon Dioxide-Based Machine Learning Model with Ventilation System and Differential Pressure Data », *in: Sensors* 23.2 (2023), p. 585, DOI: 10.3390/s23020585.
- Kim, J. et al., « Occupant comfort and behavior: High-resolution data from a 6-month field study of personal comfort systems with 37 real office workers », *in: Building and Environment* 148 (2019), pp. 348–360, DOI: 10.1016/j.buildenv.2018.11.012.
- Kim, J.-Y. and S.-B. Cho, « Explainable prediction of electric energy demand using a deep autoencoder with interpretable latent space », *in: Expert Systems with Applications* 186 (2021), p. 115842, DOI: 10.1016/j.eswa.2021.115842.
- Kim, M., I. Konstantzos, and A. Tzempelikos, « Real-time daylight glare control using a low-cost, window-mounted HDRI sensor », *in: Building and Environment* 177 (2020), p. 106912, DOI: 10.1016/j.buildenv.2020.106912.
- Kim, S. et al., « Development of a Consecutive Occupancy Estimation Framework for Improving the Energy Demand Prediction Performance of Building Energy Modeling Tools », *in: Energies* 12 (2019), p. 433, DOI: 10.3390/en12030433.
- Kim, Y.-S. and J. Srebric, « Impact of occupancy rates on the building electricity consumption in commercial buildings », *in: Energy and Buildings* 138 (2017), pp. 591–600, DOI: 10.1016/j.enbuild.2016.12.056.
- Kirkpatrick, S., C.D. Gelatt Jr., and M.P. Vecchi, « Optimization by Simulated Annealing », *in: Science* 220.4598 (1983), pp. 671–680.
- Klein, L. et al., « Coordinating occupant behavior for building energy and comfort management using multi-agent systems », *in: Automation in Construction* 22 (2012), pp. 525–536, DOI: 10.1016/j.autcon.2011.11.012.
- Klepeis, N.E. et al., « The national human activity pattern survey (NHAPS): a resource for assessing exposure to environmental pollutants », *in: J. Expo. Anal. Env. Epid.* 11.3 (2001), pp. 231–252, DOI: 10.1038/sj.jea.7500165.

-
- Koklu, M. and K. Tutuncu, « Tree based classification methods for occupancy detection », *in: IOP Conference Series: Materials Science and Engineering* 675 (2019), p. 012032, DOI: 10.1088/1757-899X/675/1/012032.
- Kondratovich, E., I. I. Baskin, and A. Varnek, « Transductive Support Vector Machines: Promising Approach to Model Small and Unbalanced Datasets », *in: Molecular informatics* 32.3 (2013), pp. 261–266, DOI: 10.1002/minf.201200135.
- Konstantakopoulos, I.C. et al., « A deep learning and gamification approach to improving human-building interaction and energy efficiency in smart infrastructure », *in: Applied Energy* 237 (2019), pp. 810–821, DOI: 10.1016/j.apenergy.2018.12.065.
- Koo, Chae-Chil, D. Seo, and Tae-Ho Kim, « Chemical Plant Explosion Accident (Cause) Analysis Using AcciMap and FRAM », *in: Korean Institute of Fire Science Engineering* (2021), DOI: 10.7731/kifse.613c0e37.
- Krishna, Chenchu Murali, Kirti Ruikar, and Kumar Neeraj Jha, « Determinants of Data Quality Dimensions for Assessing Highway Infrastructure Data Using Semiotic Framework », *in: Energy and Buildings*, vol. 13, 4, MDPI, 2023, p. 944.
- Labeodan, T. et al., « Experimental evaluation of the performance of chair sensors in an office space for occupancy detection and occupancy-driven control », *in: Energy and Buildings* 111 (2016), pp. 195–206, DOI: 10.1016/j.enbuild.2015.11.054.
- Lage, I. et al., « Human-in-the-Loop Interpretability Prior », *in: 32nd Conference on neural information processing systems (NIPS 2018)*, 2018, 13 pages, URL: <https://arxiv.org/abs/1805.11571v2>.
- Lapiente, C.S. et al., « Long-Term Assessment of a Set of CO₂ Concentration Sensors in an In-Use Office Building », *in: Sensors* 22.23 (2022), p. 9403, DOI: 10.3390/s22239403.
- Larson, Cinnamon B, David C Zimmerman, and Edward L Marek, « A comparison of modal test planning techniques: excitation and sensor placement using the NASA 8-bay truss », *in: Proceedings of the 12th International Modal Analysis* 2251 (1994), p. 205.
- Lee, Namkyoung and Dongsoo Han, « Magnetic indoor positioning system using deep neural network », *in: 2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2017, pp. 1–8.
- Li, H., T. Hong, and M. Sofos, « An inverse approach to solving zone air infiltration rate and people count using indoor environmental sensor data », *in: Energy and Buildings* 198 (2019), pp. 228–242, DOI: 10.1016/j.enbuild.2019.06.008.

-
- Li, T. et al., « A hierarchical object oriented Bayesian network-based fault diagnosis method for building energy systems », *in: Applied Energy* 306.B (2022), p. 118088, DOI: 10.1016/j.apenergy.2021.118088.
- Li, X. et al., « Improved Adaptive Multi-Objective Particle Swarm Optimization of Sensor Layout for Shape Sensing with Inverse Finite Element Method », *in: Sensors* 22.14 (2022), p. 5203, DOI: 10.3390/s22145203.
- Li, X.H. and S.H. Hong, « User-expected price-based demand response algorithm for a home-to-grid system », *in: Energy* 64 (2014), pp. 437–449, DOI: 10.1016/j.energy.2013.11.049.
- Li, Y. and Z. O’Neill, « An innovative fault impact analysis framework for enhancing building operations », *in: Energy and Buildings* 199 (2019), pp. 311–331, DOI: 10.1016/j.enbuild.2019.07.011.
- Li, Z. and B. Dong, « Short term predictions of occupancy in commercial buildings—Performance analysis for stochastic models and machine learning approaches », *in: Energy and Buildings* 158 (2018), pp. 268–281, DOI: 10.1016/j.enbuild.2017.09.052.
- Li, Zeyan et al., « Generic and Robust Localization of Multi-Dimensional Root Causes », *in: ISSRE 2019*, 2019.
- Lima, W.S. et al., « User activity recognition for energy saving in smart home environment », *in: IEEE Symposium on Computers and Communication (ISCC)*, 2015, pp. 751–757, DOI: 10.1109/ISCC.2015.7405604.
- Limongelli, Maria Paola et al., « Optimal sensor placement for early warning systems in landslide risk management. Application to the town of Sorrento (Italy) », *in: Natural Hazards and Earth System Sciences Discussions* 3.5-6 (2003), pp. 3039–3064.
- Lin, Fred et al., « Fast Dimensional Analysis for Root Cause Investigation in a Large-Scale Service Environment », *in: arXiv preprint arXiv:1911.01225* (2019).
- Lin, Q. et al., « idice: problem identification for emerging issues », *in: 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*, 2016.
- Liu, X.-Y., J. Wu, and Z.-H. Zhou, « Exploratory Undersampling for Class-Imbalance Learning », *in: IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.2 (2009), pp. 539–550, DOI: 10.1109/TSMCB.2008.2007853.
- Liu, Xiao et al., « Self-Supervised Learning: Generative or Contrastive », *in: IEEE Transactions on Knowledge and Data Engineering* 35.1 (2023), Date of Publication: 22 June 2021, pp. 857–876, DOI: 10.1109/TKDE.2021.3090866.

-
- Liu, Z., J. Zhang, and L. Geng, « An Intelligent Building Occupancy Detection System Based on Sparse Auto-Encoder », *in: 2017 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, 2017, pp. 17–22, DOI: 10.1109/WACVW.2017.10.
- Lloyd, Philip J, « The role of energy in development », *in: Journal of Energy in Southern Africa* 28.1 (2017), p. 5.
- Löhner, Rainald and Fernando Camelli, « Optimal placement of sensors for contaminant detection based on detailed 3D CFD simulations », *in: Engineering computations* 22.3 (2005), pp. 260–273.
- Longo, E., A.E.C. Redondi, and M. Cesana, « Accurate occupancy estimation with WiFi and bluetooth/BLE packet capture », *in: Computer Networks* 163 (2019), p. 106876, DOI: 10.1016/j.comnet.2019.106876.
- Lu, X. et al., « A novel simulation-based framework for sensor error impact analysis in smart building systems: a case study for a demand-controlled ventilation system », *in: Applied Energy* 263 (2020), p. 114638, DOI: 10.1016/j.apenergy.2020.114638.
- Lu, X. et al., « Extracting typical occupancy schedules from social media (TOSSM) and its integration with building energy modeling », *in: Building Simulation* 14 (2020), pp. 25–41, DOI: 10.1007/s12273-020-0637-y.
- Lu, X. et al., « Robust occupancy inference with commodity WiFi », *in: IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2016, pp. 1–8, DOI: 10.1109/WiMOB.2016.7763228.
- Luo, X. et al., « Abnormal activity detection using pyroelectric infrared sensors », *in: Sensors* 16.6 (2016), p. 822, DOI: 10.3390/s16060822.
- Luo, Yonghong et al., « Multivariate Time Series Imputation with Generative Adversarial Networks », *in: Advances in Neural Information Processing Systems*, vol. 31, 2018, p. 12, URL: <https://papers.nips.cc/paper/2018/hash/96b9bff013acedfb1d140579e2fbeb63-Abstract.html>.
- Ma, Y. et al., « Investigation of Smart Home Energy Management System for Demand Response Application », *in: Frontiers in Energy Research* 9 (2021), p. 772027, DOI: 10.3389/fenrg.2021.772027.
- Maasoumy, M. et al., « Co-design of control algorithm and embedded platform for building HVAC systems », *in: 2013 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2013, pp. 61–70.

-
- Machorro-Cano, I. et al., « HEMS-IoT: A Big Data and Machine Learning-Based Smart Home System for Energy Saving », *in: Energies* 13.5 (2020), p. 1097, DOI: 10.3390/en13051097.
- Manno, A., E. Martelli, and E. Amaldi, « A Shallow Neural Network Approach for the Short-Term Forecast of Hourly Energy Consumption », *in: Energies* 15 (2022), p. 958, DOI: 10.3390/en15030958.
- Mar, C.M. del et al., *Comfort Control in Buildings*, Springer-Verlag London, 2014.
- Marikyan, D., S. Papagiannidis, and E. Alamanos, « A systematic review of the smart home literature: A user perspective », *in: Technological Forecasting and Social Change* 138 (2019), pp. 139–154, DOI: 10.1016/j.techfore.2018.08.015.
- Martani, Claudio et al., « ENERNET: Studying the dynamic relationship between building occupancy and energy consumption », *in: Energy and Buildings* 47 (2012), pp. 584–591, DOI: 10.1016/j.enbuild.2011.12.037.
- « ENERNET: Studying the dynamic relationship between building occupancy and energy consumption », *in: Energy and Buildings* 47 (2012), pp. 584–591.
- Martinez, S. et al., « Root cause analysis of the corrosion-related coiled tubing failure », *in: Journal of Engineering Science and Technology* (2022), DOI: 10.5599/jese.1280.
- Mashal, I., A. Shuhaiber, and M. Daoud, « Factors influencing the acceptance of smart homes in Jordan », *in: International Journal of Electronic Marketing and Retailing* 11.2 (2020), pp. 113–142, DOI: 10.1504/IJEMR.2020.106842.
- Masoso, Otieno T and Louis J Grobler, « The dark side of occupants’ behaviour on building energy use », *in: Energy and Buildings* 42.2 (2010), pp. 173–177, DOI: 10.1016/j.enbuild.2009.08.009.
- McInnes, L. et al., « UMAP: Uniform Manifold Approximation and Projection », *in: Journal of Open Source Software* 3.29 (2020), p. 861, DOI: 10.21105/joss.00861.
- Mechri, H.E., A. Capozzoli, and V. Corrado, « Use of the ANOVA approach for sensitive building energy design », *in: Applied Energy* 87.10 (2010), pp. 3073–3083, DOI: 10.1016/j.apenergy.2010.04.001.
- Medhat, Walaa, Ahmed Hassan Yousef, and Hoda Korashy Mohamed, « Combined Algorithm for Data Mining using Association rules », *in: arXiv preprint arXiv:1410.1343* (2014).
- Mendes, T. et al., « Anomaly Detection of Consumption in Hotel Units: A Case Study Comparing Isolation Forest and Variational Autoencoder Algorithms », *in: Applied Sciences* 13.1 (2023), p. 314, DOI: 10.3390/app13010314.

-
- Meng, Y.-B et al., « Real-time dynamic estimation of occupancy load and an air-conditioning predictive control method based on image information fusion », *in: Building and Environment* 173 (2020), p. 106741, DOI: 10.1016/j.buildenv.2020.106741.
- Meo, M. and G. Zumpano, « On the optimal sensor placement techniques for a bridge structure », *in: Engineering Structures* 27.10 (2005), pp. 1488–1497, DOI: 10.1016/j.engstruct.2005.03.015.
- Meyn, S. et al., « A sensor-utility-network method for estimation of occupancy in buildings », *in: Proceedings of the 48th IEEE Conference on Decision and Control*, 2009, pp. 1494–1500.
- Minoli, D., K. Sohraby, and B. Occhiogrosso, « IoT considerations, requirements, and architectures for smart buildings—energy optimization and next-generation building management systems », *in: IEEE Internet of Things Journal* 4.1 (2017), pp. 269–283, DOI: 10.1109/JIOT.2017.2647881.
- Misra, I., C. L. Zitnick, and M. Hebert, « Shuffle and learn: unsupervised learning using temporal order verification », *in: ECCV 2016*, 2016.
- Missaoui, R. et al., « Managing Energy Smart Homes According to Energy Prices: Analysis of a Building Energy Management System », *in: Energy and Buildings* 71 (2014), pp. 155–167, DOI: 10.1016/j.enbuild.2013.12.018.
- Moghimi, S. et al., « Building energy index and end-use energy analysis in large-scale hospitals—case study in Malaysia », *in: Energy Efficiency* (2013), DOI: 10.1007/s12053-013-9221-y.
- Mohammadabadi, A., S. Rahnama, and A. Afshari, « Indoor Occupancy Detection Based on Environmental Data Using CNN-XGboost Model: Experimental Validation in a Residential Building », *in: Sustainability* 14.21 (2022), p. 14644, DOI: 10.3390/su142114644.
- Moradzadeh, A. et al., « Performance Evaluation of Two Machine Learning Techniques in Heating and Cooling Loads Forecasting of Residential Buildings », *in: Applied Sciences* 10 (2020), p. 3829, DOI: 10.3390/app10113829.
- Nagarathinam, S. et al., « Centralized management of HVAC energy in large multi-AHU zones », *in: Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, 2015, pp. 157–166.
- Nanda, Bayu Bisma and W. Sulistiyowati, « Minimize Defects in 5 Liters Jerry Cans by Using Statistical Quality Control and Root Cause Analysis », *in: Proxima* 4.2 (2021), DOI: 10.21070/proxima.v4i2.1302.

-
- Narayanan, S. et al., *A Wireless Platform for Energy Efficient Building Control Retrofits*, tech. rep., United Technologies Research Center East Hartford CT, 2012, p. 123.
- Naser, A. et al., « Heat-Map Based Occupancy Estimation Using Adaptive Boosting », *in: 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2020, pp. 1–7, DOI: 10.1109/FUZZ48607.2020.9177685.
- Navarro, R.C. et al., « Indoor occupancy estimation for smart utilities: A novel approach based on depth sensors », *in: Building and Environment* 222 (2022), p. 109406, DOI: 10.1016/j.buildenv.2022.109406.
- Neale, A., M. Kummert, and M. Bernier, « Discriminant analysis classification of residential electricity smart meter data », *in: Energy and Buildings* 258 (2022), p. 111823, DOI: 10.1016/j.enbuild.2021.111823.
- Newsham, G.R. and B.J. Birt, « Building-level occupancy data to improve ARIMA-based electricity use forecasts », *in: Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building (BuildSys '10)* (2010), pp. 13–18, DOI: 10.1145/1878431.1878435.
- Ngo, N.-T. et al., « Developing a hybrid time-series artificial intelligence model to forecast energy use in buildings », *in: Scientific Reports* 12 (2022), p. 15775, DOI: 10.1038/s41598-022-19935-6.
- Noroozi, M. and P. Favaro, « Unsupervised learning of visual representations by solving jigsaw puzzles », *in: ECCV 2016*, 2016.
- Novikova, E., M. Bestuzhev, and A. Shorov, « The Visualization-Driven Approach to the Analysis of the HVAC Data », *in: Intelligent Distributed Computing XIII. IDC 2019. Studies in Computational Intelligence*, vol. 868, 2020, p. 64, DOI: 10.1007/978-3-030-32258-8_64.
- Okafor, Nwamaka U. and Declan T. Delaney, « Missing Data Imputation on IoT Sensor Networks: Implications for on-Site Sensor Calibration », *in: IEEE Sensors Journal* 21.20 (2021), pp. 22833–22845, DOI: 10.1109/JSEN.2021.3105442.
- Okonkwo, S. and Z. H. Mshelia, « Estimating Aboveground Biomass Using Allometric Models And Adaptive Learning Rate Optimization Algorithms », *in: Journal of Applied Sciences and Environmental Management*, 2021, DOI: 10.4314/jasem.v25i7.6.
- Oldewurtel, F. et al., « Energy efficient building climate control using stochastic model predictive control and weather predictions », *in: American Control Conference (ACC), 2010*, 2010, pp. 5100–5105.

-
- Oldewurtel, F. et al., « Use of model predictive control and weather forecasts for energy efficient building climate control », *in: Energy and Buildings* 45.0 (2012), pp. 15–27.
- Oniga, S. and J. Sütő, « Human activity recognition using neural networks », *in: Proceedings of the 2014 15th International Carpathian Control Conference (ICCC)*, 2014, pp. 403–406, DOI: 10.1109/CarpathianCC.2014.6843636.
- Oord, Aäron van den et al., « Representation Learning with Contrastive Predictive Coding », *in: ArXiv* abs/1807.03748 (2018).
- Osegueda, R A, C J Carrasco, and R Meza, « Modal strain energy distribution method to localize and quantify damage », *in: FAST Center for Structural Integrity of Aerospace Systems*, 1997.
- Osman, Muhammad S., Adnan M. Abu-Mahfouz, and Philip R. Page, « A Survey on Data Imputation Techniques: Water Distribution System as a Use Case », *in: IEEE Access* 6 (2018), pp. 63279–63291, DOI: 10.1109/ACCESS.2018.2877269.
- Padula, S, D Palumbo, and R Kincaid, « Optimal sensor/actuator locations for active structural acoustic control », *in: 39th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and Exhibit*, 1998, p. 1865.
- Pan, S. et al., « Cluster analysis for occupant-behavior based electricity load patterns in buildings: A case study in Shanghai residences », *in: Building Simulation* 10 (2017), pp. 889–898, DOI: 10.1007/s12273-017-0377-9.
- Paone, Antonio, Michele Munafò, and Enrico Fabrizio, « The impact of occupant behavior on building energy demand: A review of the literature », *in: Renewable and Sustainable Energy Reviews* 82 (2018), pp. 2144–2160, DOI: 10.1016/j.rser.2017.10.063.
- Papadopoulou, M. et al., « Evaluating predictive performance of sensor configurations in wind studies around buildings », *in: Advanced Engineering Informatics* 30.2 (2016), pp. 127–142, DOI: 10.1016/j.aei.2016.02.004.
- Paradiso, F. et al., « ANN-based appliance recognition from low-frequency energy monitoring data », *in: 2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, 2013, pp. 1–6, DOI: 10.1109/WoWMoM.2013.6583496.
- Parise, A. et al., « Prophet model for forecasting occupancy presence in indoor spaces using non-intrusive sensors », *in: AGILE GIScience Ser. 2.9* (2021), DOI: 10.5194/agile-giss-2-9-2021.
- Parliament, European, *EU policy: Revised Energy Performance of Buildings Directive (EPBD)*, Technical Report, EUR-Lex - 32018L0844 - EN, European Parliament, 2018.

-
- Parzinger, M. et al., « Comparison of different training data sets from simulation and experimental measurement with artificial users for occupancy detection — Using machine learning methods Random Forest and LASSO », *in: Building and Environment* 223 (2022), DOI: 10.1016/j.buildenv.2022.109313.
- Pathak, Deepak et al., « Context Encoders: Feature Learning by Inpainting », *in: CVPR 2016*, 2016.
- Pattison Tuohy, J., « The CSA is addressing smart home data privacy, and it's about damn time », *in: The Verge* (Feb. 22, 2023), URL: <https://www.theverge.com/2023/2/22/23610081/smart-home-matter-data-privacy-certification-csa>.
- Pazhooesh, Mehdi, Zoya Pourmirza, and Sara Walker, « A Comparison of Methods for Missing Data Treatment in Building Sensor Data », *in: 2019 IEEE 7th International Conference on Smart Energy Grid Engineering (SEGE)*, 2019, pp. 255–259, DOI: 10.1109/SEGE.2019.8859963.
- Pedro, Gonçalo and João Gomes, « Optimal Sensor Placement for Building Energy Performance Monitoring », *in: Energy Procedia* 152 (2018), pp. 1005–1010, ISSN: 1876-6102, DOI: <https://doi.org/10.1016/j.egypro.2018.09.238>, URL: <http://www.sciencedirect.com/science/article/pii/S187661021832422X>.
- Peng, Jiaxuan et al., « Optimal sensor placement for landslide monitoring using wireless sensor networks: a case study in Three Gorges Reservoir Area, China », *in: Landslides* 15.9 (2018), pp. 1783–1795.
- Persson, M. and L. Rudenius, « Anomaly detection and fault localization an automated process for advertising systems », MA thesis, Göteborg: Chalmers University of Technology, 2018.
- Pettersen, Ida Nilstad et al., « Ambitions at work: Professional practices and the energy performance of non-residential buildings in Norway », *in: Energy Research & Social Science* 32 (2017), pp. 112–120, DOI: 10.1016/j.erss.2017.02.013.
- Prabhakaran, K. et al., « Explainable K-Means Clustering for Occupancy Estimation », *in: Procedia Computer Science*, vol. 203, 2022, pp. 326–333, DOI: 10.1016/j.procs.2022.07.041.
- Qolomany, B. et al., « Role of Deep LSTM Neural Networks And Wi-Fi Networks in Support of Occupancy Prediction in Smart Buildings », *in: 2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data*

-
- Science and Systems (HPCC/SmartCity/DSS)*, 2017, pp. 50–57, DOI: 10.1109/HPCC-SmartCity-DSS.2017.7.
- Qureshi, Muhammad Atif et al., « Optimal sensor placement for intrusion detection in wireless sensor networks using genetic algorithms. In Proceedings of the International Conference on Wireless Communications and Mobile Computing (IWCMC) », *in: IEEE Xplore Digital Library Proceedings of the International Conference on Wireless Communications and Mobile Computing (IWCMC)*, 2007, pp. 1–6.
- Rahman, H. and H. Han, « Bayesian estimation of occupancy distribution in a multi-room office building based on CO2 concentrations », *in: Building Simulation* 11 (2018), pp. 575–583, DOI: 10.1007/s12273-017-0413-9.
- Raykov, Y.P. et al., « Predicting room occupancy with a single passive infrared (PIR) sensor through behavior extraction », *in: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*, 2016, pp. 1016–1027, DOI: 10.1145/2971648.2971746.
- Robillart, M., « Étude de stratégies de gestion en temps réel pour des bâtiments énergétiquement performants », Thesis, École nationale supérieure des mines de Paris, 2015.
- Roth, K.W. et al., *Energy Consumption Characteristics of Commercial Building HVAC Systems - Volume III: Energy Savings Potential*, Technical Report, 2002.
- Rutar, Danaja et al., « Differentiating between Bayesian parameter learning and structure learning based on behavioural and pupil measures », *in: PLOS ONE* N/A (2023), N/A, DOI: 10.1371/journal.pone.0270619.
- Es-Sabar, Ahmed, « Prétraitement des données mesurées et traitement des données manquantes pour des capteurs de bâtiments connectés », Master thesis, Polytech Angers, 2022.
- Es-Sabar, Ahmed et al., « Traitement des données manquantes pour des capteurs de bâtiments connectés », *in: 2022*, URL: <https://hal.science/hal-03687624>.
- Saeed, Aaqib, Tanir Ozcelebi, and Johan Lukkien, « Multi-task Self-Supervised Learning for Human Activity Detection », *in: arXiv preprint arXiv:1907.11879* (2019).
- Salimi, S., Z. Liu, and A. Hammad, « Occupancy prediction model for open-plan offices using real-time location system and inhomogeneous Markov chain », *in: Building and Environment* 152 (2019), pp. 1–16, DOI: 10.1016/j.buildenv.2019.01.052.
- Santiago, G. et al., « Audio Feature Engineering for Occupancy and Activity Estimation in Smart Buildings », *in: Electronics* 10.2599 (2021), DOI: 10.3390/electronics10212599.

-
- Sardianos, C. et al., « Rehab-c: Recommendations for energy habits change », *in: Future Generation Computer Systems* 112 (2020), pp. 394–407, DOI: 10.1016/j.future.2020.05.041.
- Scherer, H. et al., « Distributed MPC for resource-constrained control systems », *in: Optimal Control Applications and Methods* 36.3 (2015), pp. 272–291, DOI: 10.1002/oca.2151.
- Serale, Gianluca et al., « Model Predictive Control (MPC) for Enhancing Building and HVAC System Energy Efficiency: Problem Formulation, Applications and Opportunities », *in: Energies* 11.3 (2018), DOI: 10.3390/EN11030631.
- Sermanet, Pierre et al., « Time-Contrastive Networks: Self-Supervised Learning from Video », *in: 2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1134–1141.
- Shakeri, Mohammad et al., « An intelligent system architecture in home energy management systems (HEMS) for efficient demand response in smart grid », *in: Energy and Buildings* 138 (2017), pp. 154–164, DOI: 10.1016/j.enbuild.2016.12.026.
- Shan, X. et al., « Evaluation of thermal environment by coupling CFD analysis and wireless-sensor measurements of a full-scale room with cooling system », *in: Sustainable Cities and Society* 45 (2019), pp. 395–405, DOI: 10.1016/j.scs.2018.12.011.
- Shannon, Claude, « A Mathematical Theory of Communication », *in: Bell System Technical Journal* 27.3 (1948), pp. 379–423.
- Sharma, P. et al., « Deep-Learning-Based Occupant Counting by Ambient RF Sensing », *in: IEEE Sensors Journal* 21.6 (2021), pp. 8564–8574, DOI: 10.1109/JSEN.2020.3045035.
- Shih, H.-C., « A robust occupancy detection and tracking algorithm for the automatic monitoring and commissioning of a building », *in: Energy and Buildings* 77 (2014), pp. 270–280, DOI: 10.1016/j.enbuild.2014.03.069.
- Shimosaka, M. and O. Saisho, « Efficient calibration for rssi-based indoor localization by bayesian experimental design on multi-task classification », *in: UbiComp '16: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016, pp. 244–249, DOI: 10.1145/2971648.2971710.
- Shimosaka, Masamichi et al., « ZigBee based wireless indoor localization with sensor placement optimization towards practical home sensing », *in: Advanced Robotics* 30.5 (2016), pp. 315–325.

-
- Shuhaiber, A. and I. Mashal, « Understanding users' acceptance of smart homes », *in: Technology in Society* 58 (2019), p. 101110, DOI: 10.1016/j.techsoc.2019.01.003.
- Slakey, D. et al., « Using simulation to improve root cause analysis of adverse surgical outcomes », *in: International Journal for Quality in Health Care* 26.2 (2014), pp. 144–150, DOI: 10.1093/intqhc/mzu011.
- Smarra, F. et al., « Data-driven model predictive control using random forests for building energy optimization and climate control », *in: Applied Energy* 226 (2018), pp. 1252–1272, DOI: 10.1016/j.apenergy.2018.02.126.
- Smart2B, *Data Quality in Buildings Energy Management*, 2021, URL: <https://smart2b-project.eu/blog/data-quality-in-buildings-energy-management/>.
- Smullen, Daniel et al., « Genetic Algorithm with Self-Adaptive Mutation Controlled by Chromosome Similarity », *in: 2014 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, Beijing, China, July 2014.
- Song, Youngrok, Sangwon Hyun, and Yun-Gyung Cheong, « Analysis of Autoencoders for Network Intrusion Detection », *in: Sensors* 21.13 (2021), This paper is an extended version of our paper published in Song, Y.; Hyun, S.; Cheong, Y.G. A Systematic Approach to Building Autoencoders for Intrusion Detection. In Silicon Valley Cybersecurity Conference, SVCC 2020, San Jose, CA, USA, 17–19 December 2020; Park, Y., Jadav, D., Austin, T., Eds.; Communications in Computer and Information Science; Springer: Cham, Switzerland, 2021; Volume 1383., p. 4294, DOI: 10.3390/s21134294.
- Soofi, A.A. and A. Awan, « Classification techniques in machine learning: applications and issues », *in: Journal of Basic and Applied Sciences* 13 (2017), pp. 459–465, ISSN: 1814-8085.
- Spitz, Clara et al., « Simulating Combined Heat and Moisture Transfer with Energy-Plus: An Uncertainty Study and Comparison with Experimental Data », *in: (Dec. 2013)*, URL: <https://www.aivc.org/resource/simulating-combined-heat-and-moisture-transfer-energyplus-uncertainty-study-and-comparison>.
- Stazi, F., F. Naspi, and M. D'Orazio, « Modelling window status in school classrooms. Results from a case study in Italy », *in: Building and Environment* 111 (2017), pp. 24–32, DOI: 10.1016/j.buildenv.2016.10.013.
- Stubbs, Norris and Sooyong Park, « Optimal sensor placement for mode shapes via Shannon's sampling theorem », *in: Computer-Aided Civil and Infrastructure Engineering* 11.6 (1996), pp. 411–419.

-
- Sulistiyowati, W., Danang Tri Handoko, and Hana Catur Wahyuni, « Implementation of Statistical Process Control Method and Root Cause Analysis on Quality of Bitter Tannin Tea Tin », *in: IOP Conference Series: Earth and Environmental Science* 519.1 (2020), p. 012041, DOI: 10.1088/1755-1315/519/1/012041.
- Sun, Yongqian et al., « HotSpot: Anomaly Localization for Additive KPIs with Multi-Dimensional Attributes », *in: IEEE Access* (2018).
- Suryanarayana, G. et al., « A data driven method for optimal sensor placement in multi-zone buildings », *in: Energy and Buildings* 243 (2021), p. 110956, DOI: 10.1016/j.enbuild.2021.110956.
- Szczurek, A., M. Maciejewska, and T. Pietrucha, « Occupancy determination based on time series of CO2 concentration, temperature and relative humidity », *in: Energy and Buildings* 147 (2017), pp. 142–154, DOI: 10.1016/j.enbuild.2017.04.080.
- Tang, C. et al., « Occupancy Detection and People Counting Using WiFi Passive Radar », *in: 2020 IEEE Radar Conference (RadarConf20)*, 2020, pp. 1–6, DOI: 10.1109/RadarConf2043947.2020.9266493.
- Tian, W. et al., « Optimization on thermostat location in an office room using the coupled simulation platform in modelica buildings library: a pilot study », *in: The 4th International Conference on Building Energy and Environment (COBEE2018)*, 2018.
- Tian, Yonglong et al., « Contrastive Multiview Coding », *in: ArXiv* abs/1906.05849 (2019).
- Tian, Z. et al., « An application of Bayesian Network approach for selecting energy efficient HVAC systems », *in: Journal of Building Engineering* 25 (2019), p. 100796, DOI: 10.1016/j.jobe.2019.100796.
- Tien, P.W. et al., « Real-time monitoring of occupancy activities and window opening within buildings using an integrated deep learning-based approach for reducing energy demand », *in: Applied Energy* 308 (2022), p. 118336, DOI: 10.1016/j.apenergy.2021.118336.
- Togashi, Eisuke, « Risk analysis of energy efficiency investments in buildings using the Monte Carlo method », *in: Journal of Building Performance Simulation* 11.6 (2018), pp. 644–656, DOI: 10.1080/19401493.2018.1523949.
- Tran, D.H., E. Sanchez, and M.H. Nazari, « Model predictive energy management for building microgrids with IoT-based controllable loads », *in: 2019 North American Power Symposium (NAPS)*, 2019, pp. 1–6, DOI: 10.1109/NAPS46382.2019.9005446.

-
- Tran, D.H. et al., « Smart building design: a framework for optimal placement of smart sensors and actuators », *in: 2019 IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, 2019, pp. 1–5, DOI: 10.1109/ISGT.2019.8791562.
- Turner, W.J.N., I.S. Walker, and J. Roux, « Peak load reductions: Electric load shifting with mechanical pre-cooling of residential buildings with low thermal mass », *in: Energy* 82 (2015), pp. 1057–1067, DOI: 10.1016/j.energy.2015.02.011.
- Tyndall, A., R. Cardell-Oliver, and A. Keating, « Occupancy Estimation Using a Low-Pixel Count Thermal Imager », *in: IEEE Sensors Journal* 16.10 (2016), DOI: 10.1109/JSEN.2016.2530824.
- Uddin, Mohammad Nyme et al., « Influence of Occupant Behavior for Building Energy Conservation: A Systematic Review Study of Diverse Modeling and Simulation Approach », *in: Buildings* 11.2 (2021), p. 41, DOI: 10.3390/BUILDINGS11020041.
- Vanus, J., O.M. Gorjani, and P. Bilik, « Novel Proposal for Prediction of CO2 Course and Occupancy Recognition in Intelligent Buildings within IoT », *in: Energies* 12.23 (2019), p. 4541, DOI: 10.3390/en12234541.
- Varlamis, I. et al., « Smart fusion of sensor data and human feedback for personalized energy-saving recommendations », *in: Applied Energy* 305 (2022), p. 117775, DOI: 10.1016/j.apenergy.2021.117775.
- Vela, A. et al., « Estimating Occupancy Levels in Enclosed Spaces Using Environmental Variables: A Fitness Gym and Living Room as Evaluation Scenarios », *in: Sensors (Basel, Switzerland)* 20.22 (2020), p. 6579, DOI: 10.3390/s20226579.
- Velickovic, Petar et al., « Deep Graph Infomax », *in: ArXiv* abs/1809.10341 (2018).
- Villa, M., B. Ferreira, and N. Cruz, « Genetic Algorithm to Solve Optimal Sensor Placement for Underwater Vehicle Localization with Range Dependent Noises », *in: Sensors* 22.19 (2022), p. 7205, DOI: 10.3390/s22197205.
- Vincenzi, L. and L. Simonini, « Influence of model errors in optimal sensor placement », *in: Journal of Sound and Vibration* 389 (2017), pp. 119–133, DOI: 10.1016/j.jsv.2016.10.033.
- Wahl, F., M. Milenkovic, and O. Amft, « A Distributed PIR-based Approach for Estimating People Count in Office Environments », *in: 2012 IEEE 15th International Conference on Computational Science and Engineering*, 2012, pp. 640–647, DOI: 10.1109/ICCSE.2012.92.

-
- Wang, J. et al., « A location-aware lifestyle improvement system to save energy in smart home », *in: 4th International Conference on Awareness Science and Technology*, 2012, pp. 109–114, DOI: 10.1109/iCAwST.2012.6469598.
- Wang, W., J. Chen, and T. Hong, « Occupancy prediction through machine learning and data fusion of environmental sensing and Wi-Fi sensing in buildings », *in: Automation in Construction* 94 (2018), pp. 233–243, DOI: 10.1016/j.autcon.2018.07.007.
- Wang, W. et al., « Linking energy-cyber-physical systems with occupancy prediction and interpretation through WiFi probe-based ensemble classification », *in: Applied Energy* 236 (2019), pp. 55–69, DOI: 10.1016/j.apenergy.2018.11.079.
- Wang, W. et al., « Occupancy prediction through Markov based feedback recurrent neural network (M-FRNN) algorithm with WiFi probe technology », *in: Building and Environment* 138 (2018), pp. 160–170, DOI: 10.1016/j.buildenv.2018.04.034.
- Wang, Xiaolong and Abhinav Gupta, « Unsupervised Learning of Visual Representations Using Videos », *in: 2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2794–2802.
- Wang, Z., T. Hong, and M.A. Piette, « Predicting plug loads with occupant count data through a deep learning approach », *in: Energy* 181 (2019), pp. 29–42, DOI: 10.1016/j.energy.2019.05.138.
- Wang, Z., Y. Wang, and R.S. Srinivasan, « A novel ensemble learning approach to support building energy use prediction », *in: Energy and Buildings* 159 (2018), pp. 109–122, DOI: 10.1016/j.enbuild.2017.10.085.
- Wang, Z., R. Yang, and L. Wang, « Multi-agent control system with intelligent optimization for smart and energy-efficient buildings », *in: IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, 2010, pp. 1144–1149, DOI: 10.1109/IECON.2010.5675530.
- Wang, Z. et al., « Fault detection based on Bayesian network and missing data imputation for building energy systems », *in: Applied Thermal Engineering* 182 (2021), p. 116051, DOI: 10.1016/j.applthermaleng.2020.116051.
- Weerakody, Philip B. et al., « A Review of Irregular Time Series Data Handling with Gated Recurrent Neural Networks », *in: Neurocomputing* 441 (2021), pp. 161–178, DOI: 10.1016/j.neucom.2021.02.046.
- Weng, Shaoyuan, Jin Gou, and Zongwen Fan, « *h*-DBSCAN: A simple fast DBSCAN algorithm for big data », *in: Asian Conference on Machine Learning*, PMLR, 2021, pp. 81–96.

-
- Wetter, M., « Simulation-Based Building Energy Optimization », PhD thesis, University of California, Berkeley, 2004.
- Wu, Jiawei et al., « Self-Supervised Dialogue Learning », *in: ACL*, 2019.
- Wu, X. et al., « A survey of human-in-the-loop for machine learning », *in: Future Generation Computer Systems* 135.C (2022), pp. 364–381, DOI: 10.1016/j.future.2022.05.014.
- Wu, Zhirong et al., « Unsupervised Feature Learning via Non-parametric Instance Discrimination », *in: CVPR 2018*, 2018.
- Xu, C. and H. Chen, « A hybrid data mining approach for anomaly detection and evaluation in residential buildings energy data », *in: Energy and Buildings* 215.9 (2020), p. 109864, DOI: 10.1016/j.enbuild.2020.109864.
- Xu, J. et al., « Clustering-based probability distribution model for monthly residential building electricity consumption analysis », *in: Building Simulation* 14 (2020), pp. 149–164, DOI: 10.1007/s12273-020-0710-6.
- Xu, Xiaojun et al., « Impact of occupant characteristics on energy consumption in residential buildings: A case study in China », *in: Energy and Buildings* 215 (2020), p. 109933.
- Xu, Youchang et al., « KPI Data Anomaly Detection Strategy for Intelligent Operation and Maintenance Under Cloud Environment », *in: Communications in Computer and Information Science* (2018), DOI: 10.1007/978-3-030-00828-4_31.
- Yamaguchi, Y. and Y. Shimoda, « A stochastic model to predict occupants' activities at home for community-/urban-scale energy demand modelling », *in: Journal of Building Performance Simulation* 10.5-6 (2017), pp. 565–581, DOI: 10.1080/19401493.2017.1336255.
- Yamaguchi, Yohei and Yoshiyuki Shimoda, « A stochastic model to predict occupants' activities at home for community-/urban-scale energy demand modelling », *in: Unknown Journal Title Unknown Volume* (2017), pp. 565–581, DOI: 10.1080/19401493.2017.1336255.
- Yan, K. et al., « Semi-supervised learning for early detection and diagnosis of various air handling unit faults », *in: Energy and Buildings* 181 (2018), pp. 75–83, DOI: 10.1016/j.enbuild.2018.10.016.
- Yan, Y.R. et al., « A sensor fault detection strategy for air handling units using cluster analysis », *in: Automation in Construction* 70 (2016), pp. 77–88, DOI: 10.1016/j.autcon.2016.06.005.

-
- Yang, J. et al., « k-Shape clustering algorithm for building energy usage patterns analysis and forecasting model accuracy improvement », *in: Energy and Buildings* 146 (2017), pp. 27–37, DOI: 10.1016/j.enbuild.2017.03.071.
- Yang, S. et al., « An adaptive robust model predictive control for indoor climate optimization and uncertainties handling in buildings », *in: Build Environ* 163 (2019), DOI: 10.1016/j.buildenv.2019.106326.
- Yang, S. et al., « Analysis of traffic state variation patterns for urban road network based on spectral clustering », *in: Advances in Mechanical Engineering* 9.9 (2017), pp. 1–11, DOI: 10.1177/1687814017723790.
- Yang, Y. et al., « A framework for occupancy prediction based on image information fusion and machine learning », *in: Building and Environment* 207.B (2021), p. 108524, DOI: 10.1016/j.buildenv.2021.108524.
- Yang, Y. et al., « Energy Consumption Patterns and Characteristics of College Dormitory Buildings Based on Unsupervised Data Mining Method », *in: Buildings* 13.3 (2023), p. 666, DOI: 10.3390/buildings13030666.
- Yang, Z. et al., « A systematic approach to occupancy modeling in ambient sensor-rich buildings », *in: SIMULATION* 90.8 (2014), pp. 960–977, DOI: 10.1177/0037549713489918.
- Yi, Ting-Hua, Hong-Nan Li, and Ming Gu, « Optimal sensor placement for structural health monitoring based on multiple optimization strategies », *in: The Structural Design of Tall and Special Buildings* 20.7 (2011), pp. 881–900.
- Yoganathan, D. et al., « Optimal sensor placement strategy for office buildings using clustering algorithms », *in: Energy and Buildings* 158 (2018), pp. 1206–1225, DOI: 10.1016/j.enbuild.2017.10.074.
- Yoon, Sungmin, « In-Situ Sensor Calibration in an Operational Air-Handling Unit Coupling Autoencoder and Bayesian Inference », *in: Energy and Buildings* 221 (2020), p. 110026, DOI: 10.1016/j.enbuild.2020.110026.
- Yoon, Sungmin and Yuebin Yu, « Extended Virtual In-Situ Calibration Method in Building Systems Using Bayesian Inference », *in: Automation in Construction* 73 (2017), pp. 20–30, DOI: 10.1016/j.autcon.2016.10.008.
- Yuan, J. et al., « Forecasting building occupancy: A temporal-sequential analysis and machine learning integrated approach », *in: Energy and Buildings* 252 (2021), p. 111362, DOI: 10.1016/j.enbuild.2021.111362.

-
- Yuan, Y. et al., « Occupancy Estimation in Buildings Based on Infrared Array Sensors Detection », *in: IEEE Sensors Journal* 20.2 (2019), pp. 1043–1053, DOI: 10.1109/JSEN.2019.2943157.
- Zhai, Xiaohua et al., « SL: Self-Supervised Semi-Supervised Learning », *in: ICCV 2019*, 2019.
- Zhang, J. et al., « Room zonal location and activity intensity recognition model for residential occupant using passive-infrared sensors and machine learning », *in: Building Simulation* 15 (2022), pp. 1133–1144, DOI: 10.1007/s12273-021-0870-z.
- Zhang, R., P. Isola, and A. A. Efros, « Colorful image colorization », *in: ECCV 2016*, 2016.
- « Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction », *in: CVPR 2017*, 2017.
- Zhang, W., Y. Wu, and J.K. Calautit, « A review on occupancy prediction through machine learning for enhancing energy efficiency, air quality and thermal comfort in the built environment », *in: Renewable and Sustainable Energy Reviews* 167 (2022), p. 112704, DOI: 10.1016/j.rser.2022.112704.
- Zhang, X. et al., « Time-dependent solar aperture estimation of a building: Comparing grey-box and white-box approaches », *in: Renew Sustain Energy Rev* 161 (2022), DOI: 10.1016/j.rser.2022.112337.
- Zhao, Z. et al., « An Optimal Power Scheduling Method for Demand Response in Home Energy Management System », *in: IEEE Transactions on Smart Grid* 4.3 (2013), pp. 1391–1400, DOI: 10.1109/TSG.2013.2251018.
- Zhou, S. et al., « Real-time Energy Control Approach for Smart Home Energy Management System », *in: Electric Power Components and Systems* 42.3-4 (2014), pp. 315–326, DOI: 10.1080/15325008.2013.862322.
- Zhou, Y. and A. Dexter, « Estimating the Size of Incipient Faults in HVAC Equipment », *in: HVACR Research* 15.1 (2009), pp. 151–163, DOI: 10.1080/10789669.2009.10390830.
- Zou, Dandan et al., « Research on network cloud equipment anomaly and root cause analysis », *in: ITU Journal: ICT Discoveries* (2022), DOI: 10.52953/tv1o2995.
- Zou, H. et al., « DeepSense: Device-Free Human Activity Recognition via Autoencoder Long-Term Recurrent Convolutional Network », *in: 2018 IEEE International Conference on Communications (ICC)* (2018), pp. 1–6, DOI: 10.1109/ICC.2018.8422895.

-
- Zou, H. et al., « Towards occupant activity driven smart buildings via WiFi-enabled IoT devices and deep learning », *in: Energy and Buildings* 177 (2018), pp. 12–22, DOI: 10.1016/j.enbuild.2018.08.010.
- Zou, H. et al., « WinLight: a WiFi-based occupancy-driven lighting control system for smart building », *in: Energy and Buildings* 158 (2018), pp. 924–938, DOI: 10.1016/j.enbuild.2017.09.001.
- Zuraimi, M.S. et al., « Predicting occupancy counts using physical and statistical CO2-based modeling methodologies », *in: Building and Environment* 123 (2017), pp. 517–528, DOI: 10.1016/j.buildenv.2017.07.027.
- Al-Zwainy, F., I. Mohammed, and I. F. Varouqa, « Diagnosing the Causes of Failure in the Construction Sector Using Root Cause Analysis Technique », *in: Journal of Engineering* (2018), DOI: 10.1155/2018/1804053.

ANNEX ON CHAPTER 1

A.1 Energy performance indicators

Eight energy performance indicators have been identified in the literature (section 1.1.1). The ones that were not defined in Chapter 1 are presented below.

The Energy Use Index (EUI), belonging to the first category of indicators above, is a common EnPI used to compare building energy performance, but it has limitations as it does not account for building characteristics, occupant behavior, weather conditions, or energy service levels (Fairey and Goldstein, 2016). (Hsien-te and Chia-ju, 2021) propose a dynamic zone EUI method for hotels in Taiwan that considers different energy zones within a hotel and their corresponding EUIs, using electricity consumption as the evaluation indicator. They also suggest using yearly occupancy rate as a modifying factor. The authors test their method on 89 hotels and show that it provides a fair and customized assessment of hotel energy performance. However, the article lacks empirical evidence on the influence of the proposed rating system on hotel owners' or managers' behavior and does not discuss its applicability to other building types or regions. Zheng et al. (2019) propose a method to evaluate overall energy consumption of buildings based on energy index obtained from different functional sectors, focusing on multifunctional buildings. Their study develops a linear regression model using monitoring data from single functional and multifunctional buildings, showing that the sub-item EUI from multifunctional buildings has lower error. However, this study only considers electricity consumption as the evaluation indicator for EUI.

Energy Conservative Index (ECI) which measures the energy conservation performance of a building by comparing its actual energy consumption with its estimated energy needs (Long et al. 2014). It is calculated by dividing the actual energy consumption by the estimated energy needs, with a value below 1 indicating higher energy conservation performance.

Energy Demand Intensity (EDI) which measures the intensity of a building's energy

demand by expressing the amount of energy required to meet its specific energy needs, such as heating, cooling, lighting, etc. It is calculated by dividing the total energy required to meet the building's demand by its surface area or volume.

The indicators mentioned above (EUI, ECI, EDI. . .) are often employed as indicators of efficiency; however, this interpretation may be flawed if one considers the definition of efficiency provided in this context. Indeed, low energy use could potentially indicate high efficiency, effective operations and maintenance (O&M) in an inefficient building, or minimal tenant demands for energy services. The Energy Efficiency Index (EEI), also known as the Building Energy Index (BEI), is a widely used indicator for assessing and comparing the energy consumption performance of buildings. The EEI has gained popularity as a universal measure for evaluating energy efficiency in buildings. Typically, the EEI is calculated as the ratio of energy input to a factor associated with the energy-using component. A study conducted by Hayati et al. (2014) on sustainable energy management practices and their effect on EEI in university buildings used the EEI model to compare energy consumption performance of buildings with a focus on the air-conditioning area as the normalizing factor. Another study by Moghimi et al. (2013) conducted a case study on the EEI of commercial buildings, which was based on the occupied air conditioning area. The measurement of EEI may vary depending on the activities that take place in a building, as the energy consumption and gross floor area size may differ. This index was used to compare end-use energy consumption in a large-scale hospital building in Malaysia. The case study was carried out at Universiti Kebangsaan Malaysia Medical Centre, where the EEI of the hospital was calculated and compared to the EEI of other hospitals to assess the level of energy usage in this hospital.

The Home Energy Rating System (HERS) rating was developed by RESNET, a non-profit organization that sets standards for home energy ratings, in 2006. This system establishes a reference house that complies with the International Energy Conservation Code of 2006 and present an ideal maximum score of 100. A net-zero energy house has a score of 0, and scores extend linearly in both directions, with lower scores indicating better energy performance. Since its inception, almost 2 million homes have been rated using the HERS system, with most of these homes being rated during construction to guide energy efficiency improvements. On the other hand, the zero Energy Performance Index (zEPI) target applies to buildings that are not covered under the scope of IECC residential buildings. This system establishes a reference building with a score of 100, which represents the average energy use of a building of the same type in the same cli-

mate zone at the turn of the millennium. Like the HERS system, a score of 0 represents zero net energy, and lower scores indicate better energy performance. Recently, ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers) has established an energy standard compliance path for its Standard 90.1 based on the zEPI score, providing a benchmark for evaluating the energy performance of commercial and other non-residential buildings.

The carbon dioxide (CO₂) emissions indicator is a crucial tool in measuring the environmental impact of buildings during their operational and construction phases. This metric provides valuable insights into the amount of CO₂ emissions generated by buildings, allowing us to assess their contribution to greenhouse gas emissions, track progress in emission reduction efforts, and identify areas where further action is needed.

A.2 Pros and Cons of different technologies of IoT technologies

Table A.1 – Description, strengths, and weaknesses of monitoring and data collecting and transferring technologies.

Technology	Communication Type	Short Description	Strengths	Weaknesses
Bluetooth	Wireless	Low-power wireless technology with support for short-range communication, widely available in many devices.	<ul style="list-style-type: none"> - Low-power wireless technology with support for short-range communication. - Widely available in many devices, such as smartphones and wearables. 	<ul style="list-style-type: none"> - Limited range and bandwidth compared to some other wireless technologies. - May not be suitable for large-scale smart home applications with numerous devices.
EnOcean	Wireless	Wireless communication technology that uses energy harvesting and ultra-low power consumption for self-powered wireless sensor networks.	<ul style="list-style-type: none"> - Self-powered and battery-free operation. - Low power consumption. - Wireless communication without the need for batteries or external power source. - Suitable for energy-efficient and maintenance-free applications. 	<ul style="list-style-type: none"> - Limited range compared to other wireless technologies. - Lower data rates. - May require specific hardware and protocols for compatibility.
Ethernet	Wired	Wired communication technology that uses Ethernet cables to connect devices in a local area network (LAN).	<ul style="list-style-type: none"> - Reliable and widely used technology for wired communication. - Supports high data rates and long distances. - Provides stable and low-latency communication. 	<ul style="list-style-type: none"> - Requires physical cabling infrastructure for installation. - May require professional installation and configuration. - Not suitable for remote or wireless communication.
KNX	Wired	Wired communication protocol for home and building automation, widely used in Europe.	<ul style="list-style-type: none"> - Wired communication protocol with high reliability and stability. - Supports a wide range of devices and applications. 	<ul style="list-style-type: none"> - Requires wiring for installation, which may increase installation cost.
LoRaWAN	Wireless	Low-power, long-range wireless protocol for IoT applications, enabling long-range communication with low power consumption.	<ul style="list-style-type: none"> - Long-range communication capability with low power consumption, suitable for IoT applications. - Scalable and cost-effective for large-scale networks. 	<ul style="list-style-type: none"> - Lower data rates compared to other wireless technologies. - Limited bandwidth for data transmission. - Requires gateway devices for connecting to the internet.

Continued on next page

Table A.1 – continued from previous page

Technology	Communication Type	Short Description	Strengths	Weaknesses
PLC	Wired/Hybrid	Wired/hybrid technology using existing power lines for communication, suitable for both data and power transmission.	<ul style="list-style-type: none"> - PLC (Powerline Communication) is a wired/hybrid technology that uses existing power lines for communication, eliminating the need for additional wiring. - Can be used for both data and power transmission. 	<ul style="list-style-type: none"> - May be affected by electrical noise and interference from other devices. - Data rates may be lower compared to some wireless technologies.
RF	Wireless	Wireless technology using radio frequency, widely used in various applications including smart home devices.	<ul style="list-style-type: none"> - RF (Radio Frequency) technology is wireless and widely used in various applications, including smart home devices. - Can operate on multiple frequency bands, allowing for flexibility in deployment. 	<ul style="list-style-type: none"> - Limited range compared to some other wireless technologies. - May be susceptible to interference in crowded environments.
RFID	Wireless	Wireless technology for short-range communication and identification purposes, with low cost and small form factor.	<ul style="list-style-type: none"> - RFID (Radio Frequency Identification) is a wireless technology used for short-range communication and identification purposes. - Low cost and small form factor make it suitable for certain smart home applications. 	<ul style="list-style-type: none"> - Limited data storage and transmission capabilities compared to other wireless technologies. - Short range may require proximity for communication.
Thread	Wireless	Wireless mesh networking protocol designed for low-power devices, based on IPv6, suitable for smart home applications.	<ul style="list-style-type: none"> - Wireless mesh networking protocol designed for low-power devices, based on IPv6. - Supports large-scale networks with high reliability and security. 	<ul style="list-style-type: none"> - Limited availability of Thread-enabled devices compared to other technologies. - May require additional configuration for interoperability with other protocols.

Continued on next page

Table A.1 – continued from previous page

Technology	Communication Type	Short Description	Strengths	Weaknesses
UPnP	Hybrid	Hybrid communication protocol for device discovery and control in local networks, widely used in smart home applications.	<ul style="list-style-type: none"> - Hybrid communication protocol for device discovery and control in local networks. - Provides interoperability and easy integration of devices from different manufacturers. 	<ul style="list-style-type: none"> - Limited to local networks, may not support remote access without additional configuration. - Security concerns due to potential vulnerabilities in device discovery and control.
WiFi	Wireless	Widely used wireless technology with high data rates and extensive range.	<ul style="list-style-type: none"> - Widely used wireless technology with high data rates and extensive range. - The IEEE 802.11 standard is supported, and there is compatibility with many devices. 	<ul style="list-style-type: none"> - Higher power consumption compared to some other wireless technologies. - Possible interference from other WiFi networks or devices.
X10	Wired/Hybrid	Wired/hybrid technology for home automation widely used in older systems.	<ul style="list-style-type: none"> - Wired/hybrid technology for home automation, widely used in older systems. - Affordable cost and compatibility with a wide range of devices. 	<ul style="list-style-type: none"> - May be less reliable and slower compared to wireless technologies. - Limited in terms of features and capabilities compared to newer technologies.
Z-Wave	Wireless	Wireless communication protocol specifically designed for smart home applications with low power consumption and extended range.	<ul style="list-style-type: none"> - Specifically designed wireless communication protocol for smart home applications, with low power consumption and extended range. - Supports IP and offers a high range but is relatively less expensive. - Uses licensed frequency band to minimize interference. 	<ul style="list-style-type: none"> - To maintain security, a certain level of expertise is necessary. - Limited in terms of bandwidth and data rates. - May require gateway devices for compatibility with other technologies.

Continued on next page

Table A.1 – continued from previous page

Technology	Communication Type	Short Description	Strengths	Weaknesses
ZigBee	Wireless	Wireless communication protocol for smart appliances with IPv6 support, cost-effective, energy-efficient, and designed for low data rates.	- The IEEE 802.15.4 standard is a wireless communication protocol that supports IPv6 and is cost-effective, energy-efficient, and designed for low data rates. It enables easy connectivity of smart devices and has a flexible structure for a high number of nodes.	- Limited available bandwidth, which may result in lower data rates. - Security concerns may be higher compared to WiFi.

A.3 Optimal sesnor location algorithms

Table A.2 – Summary of optimal sensor location algorithms

Methodology	Example Method	Main Features	Advantage	Limitation	Application	Reference
Optimization	Genetic Algorithm	Global search, adaptable heuristics	Robust, handles complex landscapes	Convergence speed, parameter tuning	Structural health	holland et al.1992
	Particle Swarm	Stochastic, population-based	Simple, easy to implement	Convergence speed, local optima	Structural health	Kennedy et al.1995
	Simulated Annealing	Exploration and exploitation, annealing	Escapes local optima, adaptable	Parameter tuning, local optima	Structural health	Kirkpatrick et al.1983
Information-based	Information Entropy	Maximizes information gain	Captures diverse information	Assumes independence	Structural health , building	Shanon 1984
	Mutual Information	Takes into account redundancy	Addresses dependencies between sensors	Computational complexity	Building	Shanon 1984
Statistical	Effective Independence	Minimizes condition number of FIM	Robust to noise, linear model	Assumes linear Gaussian model	Control systems, finance	Kammer et al.1991
	Maximum Likelihood	Maximizes likelihood of observed data	Considers underlying model, well-established	Model-dependent, computational complexity	Structural health	Fisher et al.1922
Model-based	Modal Kinetic Method	Modal decomposition, kinetic energy	Applicable to large-scale systems	Limited to linear, undamped systems	Structural health	Larson et al.1994
	Drive Point Residue	Frequency response, residues	Works for a wide range of frequencies	Limited to linear systems	Structural health	Doebing et al.1996
	QR Decomposition	Orthogonal decomposition, linear independence	Handles ill-conditioned problems	Assumes linear Gaussian model	Structural health	Gander et al.2012
	Guyan Reduction	Reduced-order models, static condensation	Fast, computationally efficient	Limited to linear, undamped systems	Structural health	Guyan et al.1965
Data-driven	Space Domain Sampling	Uniform sampling in space domain	Easy to implement, intuitive	Assumes uniformity, ignores local features	Structural health	Stubbs et al.1996
	Cluster Algorithm	Groups similar data points, centroid	Reduces redundancy, simplifies data	Requires distance metric, pre-defined	Structural health ,Building	Yoganathan et al.2018
Empirical	Experience Method	Based on expert knowledge, trial-and-error	Utilizes domain knowledge, adaptable	Subjective, may lack generalization	Building, Geological disaster	lohner et al.2005
	Statistical Method	Data-driven, distribution fitting	Empirical, accounts for uncertainties	Assumes underlying statistical model	Building	Shimosaka et al.2016

A.4 Taxonomies for occupant behavior

Table A.3 – Example of taxonomy for occupant behavior

Taxonomy	Features	Examples
Activity-based taxonomy	Personal activities	Sleeping, cooking, watching TV, reading, working...
	Professional activities	Facilitating/attending a meeting, giving/-following training, using a computer, designing working documents...
	Shared activities	Socializing, hosting guests...
	Building-related activities	Maintenance, cleaning...
	Building control-related activities	Adjusting thermostat, opening/closing windows...
Location-based taxonomy	Indoor spaces	Living room, bedroom, kitchen, bathroom...
	Outdoor spaces	Balcony, terrace, garden...
Time-based taxonomy	Daily routines	Waking up, going to work, coming back home, going to bed
	Weekly routines	Grocery shopping, doing laundry...
	Seasonal routines	Turning on/off heating or cooling systems, opening/closing windows...
Behavioral patterns-based taxonomy	Predictable behavior	Regular daily routines and activities
	Unpredictable behavior	Irregular activities, unexpected visitors...
Preference-based taxonomy	Comfort preferences	Preferred temperature, lighting...
	Personal preferences	Preferred furniture, decorations...



Table A.4 – Other example of taxonomy for occupant behavior

Related behaviors	Short description
Adaptive	These behaviors are based on the adaptation of occupants to their environment, including adjusting to lighting conditions, noise levels, and air quality.
Comfort-related	Close to the previous one, these include actions that occupants take to regulate their comfort levels such as adjusting thermostats, opening windows, and using fans.
Energy-related	These behaviors are related to energy use in a building such as turning off lights, unplugging electronics, and using energy-efficient appliances.
Environmentally conscious	These behaviors are related to occupant concern for the environment such as recycling, using green products, and conserving water.
Health-related	These behaviors are related to occupant health such as exercising, taking medication, and using medical equipment.
Information-related	These involve information seeking or information sharing behaviors such as searching for information on the internet, using social media, and participating in surveys or focus groups.
Movement-related	These behaviors involve the movement of occupants within a building such as walking, running, and using stairs or elevators.
Personal habits	These include individual behaviors such as smoking, drinking, and sleeping.
Safety-related	These behaviors are related to safety measures taken by occupants such as locking doors and windows, turning off appliances, and responding to fire alarms.
Social-related	These behaviors involve interactions with other occupants or visitors such as talking, playing music, and hosting gatherings.
Task-related	These behaviors are related to specific tasks that occupants perform in a building such as cooking, cleaning, and doing laundry.

A.5 Strengths and weakness of occupancy detection sensors

Table A.5 – Strengths and weaknesses of some sensors used for building occupancy estimation.

Sensor	Strengths	Weaknesses
Audio (extraction)	- Audio sensors can detect occupancy without requiring any invasion of privacy. - They can detect a range of different sounds, from footsteps to conversations, allowing them to accurately detect occupancy in various types of spaces. - They are often less expensive than other types of occupancy sensors.	- Audio sensors can potentially record conversations and other sensitive information. - Ambient noise can interfere with the accuracy of audio sensors. - They typically have a limited range and may not be suitable for monitoring large spaces or detecting occupancy in multiple rooms simultaneously.
Camera	- Cameras provide visual information that can be used to detect and track occupants. - They can provide information about occupancy and activities, such as posture, gestures, and facial expressions. - They can be used for security purposes in addition to occupancy detection.	- Cameras can be invasive and raise privacy concerns. - They can be expensive and require additional processing power to analyze the visual data. - They require a line-of-sight, so they may miss detections if occupants are obstructed by objects or if the camera is in a blind spot.
Chair	- Chair sensors can accurately detect occupancy in a given space. - They do not require any physical contact or wearable devices. - They are relatively inexpensive compared to other occupancy detection methods. - They require minimal power to operate.	- Chair sensors are limited to the area where the sensor is placed and may not provide a comprehensive view of the entire space.
CO ₂	- CO ₂ sensors are inexpensive and compact. - They are non-invasive and easy to install. - They can provide rough occupancy estimations.	- They present limited accuracy when estimating a large number of people. - The dynamic response of CO ₂ sensors is slow.
Depth	- Depth sensors can capture spatial information about a room or space, including the number of people, their position, and movement. - They are not affected by ambient lighting conditions, unlike cameras or light sensors. - They can work even in complete darkness, making them suitable for 24/7 monitoring.	- Depth sensors may have difficulty distinguishing between people and objects that are similar in shape or size, such as furniture or large bags. - They may have difficulty detecting people partially or completely obstructed by objects. - Depth sensors are typically more expensive than some other types of occupancy sensors.

Continued on next page

Table A.5 – Continued from previous page

Sensor	Strengths	Weaknesses
Doppler Radar	<ul style="list-style-type: none"> - Doppler radar sensors can detect motion at ranges of several meters, making them suitable for use in larger spaces. - They can operate effectively in a range of environmental conditions, including darkness, dust, smoke, and temperature extremes. - Unlike PIR sensors, Doppler radar sensors can detect multiple individuals in a single zone. 	<ul style="list-style-type: none"> - Doppler radar sensors can be relatively expensive compared to other types of sensors. - They require power to operate. - They rely on detecting changes in motion to sense occupancy, which means that they may not be suitable for use in environments where movement patterns are unpredictable or irregular.
Door and Window Opening/Closing	<ul style="list-style-type: none"> - These sensors provide valuable information about the opening and closing of doors and windows, allowing for accurate occupancy estimation. - By detecting open windows or doors, these sensors can trigger actions such as adjusting heating or cooling systems, resulting in energy savings. - Door and window sensors contribute to security systems by alerting occupants to potential unauthorized access or open entry points. 	<ul style="list-style-type: none"> - Mounting door and window sensors may require careful placement and alignment to ensure accurate detection. - Sensors are limited to monitoring specific entry points, and additional sensors may be needed to cover larger areas or multiple access points. - Factors such as sensor sensitivity, environmental conditions, and human behavior can lead to false readings, impacting the accuracy of occupancy estimation.
Electric Power Consumption	<ul style="list-style-type: none"> - These sensors can be installed without disrupting the electrical infrastructure, as they typically rely on current and voltage measurements. - They provide detailed information about energy usage patterns, allowing for better energy management and optimization. - By analyzing the power consumption signatures, these sensors can help identify specific appliances or devices contributing to overall energy consumption. 	<ul style="list-style-type: none"> - Proper installation and connection to electrical circuits may require professional expertise, especially in larger-scale deployments. - Monitoring electric power consumption raises privacy issues, as it provides insights into occupants' activities and appliance usage patterns. - They focus solely on energy monitoring and may not capture other occupancy-related information or behaviors.
Humidity	<ul style="list-style-type: none"> - Humidity sensors are inexpensive and widely available. - They are non-invasive and easy to install. - They can be used to detect changes in occupancy based on moisture generated by occupants. 	<ul style="list-style-type: none"> - They present limited accuracy when estimating occupancy count. - Humidity can be influenced by factors other than occupancy, such as outdoor humidity, HVAC system, and ventilation.
Light	<ul style="list-style-type: none"> - Light sensors are non-intrusive and non-invasive. - They can be used to estimate occupancy in areas where other sensors may not be suitable, such as areas with high ceilings or difficult to access places. - They can be used to detect changes in occupancy patterns over time. 	<ul style="list-style-type: none"> - Light sensors may not be suitable for detecting occupancy during daylight hours or in areas with large windows or skylights that allow natural light to enter the space. - Changes in lighting levels due to factors such as cloud cover, shadows, reflections, and other visual obstructions can affect the accuracy of occupancy detection.

Continued on next page

Table A.5 – Continued from previous page

Sensor	Strengths	Weaknesses
PIR	<ul style="list-style-type: none"> - PIR sensors are widely available and inexpensive. - They detect infrared radiation, which is emitted by humans and animals. - They have a fast response time and can detect people moving quickly. 	<ul style="list-style-type: none"> - PIR sensors can be affected by changes in temperature and humidity, leading to false detections or missed detections. - They are not effective in detecting static occupants, such as someone sitting at a desk.
Temperature	<ul style="list-style-type: none"> - Temperature sensors are inexpensive and widely available. - They are non-invasive and easy to install. - They can be used to detect changes in room occupancy based on body heat. 	<ul style="list-style-type: none"> - They present limited accuracy when estimating occupancy count. - Temperature can be influenced by factors other than occupancy, such as outdoor temperature, HVAC system, and sunlight.
Ultrasound	<ul style="list-style-type: none"> - Ultrasonic sensors are non-invasive and do not require line-of-sight, so they can detect occupants even if they are obstructed by objects. - They have a fast response time and can detect people moving quickly. 	<ul style="list-style-type: none"> - Ultrasonic sensors can be affected by changes in temperature and humidity, leading to false detections or missed detections. - They can be affected by acoustic noise and echoes, leading to false detections.
Volatile Organic Compound (VOC)	<ul style="list-style-type: none"> - VOC sensors provide real-time feedback on the presence of volatile organic compounds, allowing for quick response and action. - These sensors can help monitor and improve the air quality in indoor environments by detecting harmful or odorous gases. - VOC levels can serve as an indirect indicator of occupancy, as human activities and behaviors often result in the release of volatile organic compounds. 	<ul style="list-style-type: none"> - VOC sensors may not be able to differentiate between different types of volatile organic compounds, leading to less precise identification of specific pollutants. - Regular calibration and maintenance are required to ensure accurate readings, as environmental factors and sensor drift can affect the performance of VOC sensors. - High-quality VOC sensors can be relatively expensive, especially if multiple sensors are needed for comprehensive coverage.

A.6 Application of artificial intelligence to detect occupancy related activities and events

Table A.6 – Examples of applications, with information on the different types of sensor data used, the methods used and their accuracy.

Type of detected activities / events	References	Sensors	Algorithm / Method	Environment	Accuracy
Occupancy presence	Arief-Ang et al. (2018)	CO ₂ data and indoor human occupancy	Seasonal-trend decomposition (STD)	An academic office and a cinema theatre	Average of 94.68 %
	Candanedo et al. (2017 a-b)	Temperature, humidity, humidity ratio, CO ₂ and light time series data	Hidden Markov Models (HMM)	Low energy residential building	Max. 90.24 %
	Chen et al. (2016)	CO ₂ , humidity, temperature and pressure levels	Support vector machine (SVM), artificial neural network (ANN), k-nearest neighbors (KNN), linear discriminant analysis (LDA) and classification and regression tree (CART)	Office buildings	Around 93 %
	D'Oca and Hong (2015)	Occupancy schedules data	Decision tree and cluster analysis	16 offices	Max. 90.53 %
	Kampeidou et al. (2021)	CO ₂ and temperature	Physics-Informed Pattern-Recognition Machine	Different types of buildings	Max. 97 %
	Kim et al. (2019b)	Temperature, illuminance, lighting power, occupancy status, relative humidity, CO ₂ , EHP energy consumption, PC energy consumption	Classification and regression tree (CART) and Support Vector Machine (SVM)	1 private office space	Average of 95 %
	Vanus et al. (2019)	Relative humidity, temperature, and CO ₂	Linear Regression, Neural Networks, and Random Tree	A laboratory	Higher than 90 %

Continued on next page

Table A.6 – continued from previous page

Type of detected activities / events	References	Sensors	Algorithm / Method	Environment	Accuracy
	Yang et al. (2014)	Twelve ambient sensor variables (temperature, CO ₂ , door status, light, etc.)	Support vector machine (SVM), k-nearest neighbors (k-NN), artificial neural network (ANN), naive Bayesian, tree augmented naive Bayesian network, Decision tree	Different types of buildings	Range 69.2-92.6 %
Occupancy number	Apostolo et al. (2021)	28 Wi-Fi Apps	Multilayer Perceptron ANN	5 floors of classrooms	RMSPE of 0.29
	Arora et al. (2015)	Power consumption, Motion detector, CO ₂ , Door contact, Window contact, Indoor temperature	Decision Tree	Office buildings	Average estimation error of 0.47 occupants
	Salimi et al. (2019)	Real-Time Locating System	Inhomogeneous Markov chain	A research laboratory	86% on average
	Sharma et al. (2021)	Wireless communication, ambient RF sensing	CNN	Residential building	82% real occupancy number
	Tang et al. (2020)	Passive Wi-Fi Radar (PWR)	CNN	Office building	Max. 98.14%
	Wang et al. (2018a)	WiFi probes	Markov based feedback recurrent neural network (M-FRNN)	Graduate student office	Max. 93.9%
	Wang et al. (2018c)	Three data sources, including environmental data, Wi-Fi data, and fused data	k-nearest neighbors (kNN), support vector machine (SVM), and artificial neural network (ANN)	Graduate student office	ANN works best for environmental data and fused data, SVM works best for Wi-Fi data
	Wang et al. (2019a)	Wi-Fi probes and indoor air temperature, relative humidity, and airflow rate	Gradient tree boosting, Random forests, Adaboost	A large office room, 200 m ²	Max. 72.7%
					Continued on next page

Table A.6 – continued from previous page

Type of detected activities / events	References	Sensors	Algorithm / Method	Environment	Accuracy
	Zou et al. (2018b)	WiFi-enabled IoT devices	Deep learning-based human activity recognition scheme (Deep-hare)	Conference room, office, and apartment	Max. 97.6%
Occupancy activities	Fang and Hu (2014)	Smart phones	Back Propagation (BP) neural network, Naïve Bayes (NB) classifier and Hidden Markov Model (HMM)	Residential building	BP neural network more accurate
	Huchuk et al. (2019)	Temperature and PIR sensors	Markov model (MM), HMM, and RNN	Single family homes	Under 80 % average
	Lee et al. (2017)	Smart phones	One-dimensional Convolutional neural network	Residential building	Max. 92.71%
	Lu et al. (2020b)	Social networks	Random Forest and XGBoost	A public museum	RMSE within 30%
	Tien et al. (2021)	Camera	CNN	Office space	Average of 92.2%

A.7 Strengths and weakness of artificial intelligence algorithms

The columns descriptions of Table A.7 on the strengths and weakness of artificial intelligence algorithms, is explained below:

Main Features — Family: Statistical methods; Machine learning method; Deep learning method; Ensemble learning method; Decision Tree and Graphical-based models.

- Primary usage: Classification; Regression; Detection; Time-series forecasting; Probabilistic Reasoning; Data generation; etc.
- Main category: Supervised, semi-supervised, unsupervised or specific to time-series.

Type of Data — Image data: for computer vision models that can capture spatial patterns and extract relevant visual features.

- Sequential data: for sequence models that can capture temporal dependencies and model long-term relationships.
- Text data: for Natural Language Processing models that can understand language structure and relationships and can be used for tasks such as text classification, text generation, or machine translation.
- Anomaly detection data: for models that can learn to reconstruct normal data and detect examples that deviate from the norm.
- Structured data: for structured data with well-defined features, that can handle categorical and numerical variables and are often used in classification and regression tasks.

Complexity — Basic or Simple Models: These are the simplest models that provide a basic approximation of the relationships between variables.

- Intermediate Models: These models have slightly higher complexity and are capable of capturing more complex relationships between input and output variables.
- Advanced or Complex Models: These models are more sophisticated and are capable of representing non-linear relationships and complex interactions between variables.
- Cutting-edge or Advanced Models: These are the most complex and powerful models, often utilizing deep architectures or advanced learning methods to

capture complex structures in the data.

Selected references: These articles were chosen from the literature consulted and selected for their pedagogical quality, completeness, representativeness or originality.

Table A.7 – Examples of models, with their main features, categories, type of data, complexity, strengths and weaknesses.

Models (alphabetical order)	Main features and applications	Main category	Best-suited data type	Complexity	Strengths	Weaknesses	Selected references
Adaptive Boosting (AdaBoost)	Ensemble learning method for classification and regression	Supervised	Structured data	Intermediate models	1. Handling high-dimensional data, 2. Mitigating overfitting, 3. Combining weak classifiers into a strong one	1. Sensitive to noisy data and outliers, 2. Potential model complexity, 3. Longer training time	Wang et al. (2019a)
Artificial Neural Network (ANN)	Deep learning method for classification and regression	Supervised	Various data types (can handle structured, sequential, and image data)	Advanced or complex models	1. Learning complex patterns, 2. Scalability to large datasets, 3. Handling various data types	1. Need for large amounts of labeled data, 2. Potential overfitting, 3. Lack of interpretability	Oniga and Sütő (2014), Zhang et al.(2022b)
Auto-encoder (AE)	Deep learning method for dimensionality reduction, feature extraction (unsupervised learning)	Semi-supervised or unsupervised	Any type of data (that can be used for representation learning and feature extraction)	Intermediate models	1. Effective in learning compressed representations or latent features from input data, 2. Capturing meaningful representations of complex patterns and correlations in the data, 3. Useful for denoising data and removing redundancy	1. Reconstruction may not be perfect, leading to some loss of information, 2. Showing sensitivity to hyperparameters and architecture design choices, 3. Requiring a large amount of training data	Liu et al. (2017)
Auto-encoder Long-term Recurrent Convolutional Network (ALRCN)	Deep learning method for regression and (anomaly) detection	Semi-supervised	Sequential data	Advanced or complex models	1. Capturing long-term dependencies, 2. Handling sequential data, 3. Extracting hierarchical features	1. Requiring a large amount of training data, 2. Longer training time, 3. Potential overfitting	Zou et al. (2018c)
Autoregressive Integrated Moving Average (ARIMA)	Statistical method for time series forecasting	Specific	Sequential data	Intermediate models	1. Modeling time series data, 2. Capturing trends and seasonality, 3. Providing interpretable forecasts	1. Assumption of linear relationships, 2. Difficulty in handling nonlinear data patterns, 3. Limited ability to capture long-term dependencies	Newsham and Birt (2010)
Bayesian Network (BN)	Graphical-based models for classification and probabilistic reasoning	Unsupervised	Structured data with dependencies between variables	Intermediate models	1. Capturing probabilistic relationships, 2. Handling uncertainty, 3. Allowing for causal reasoning	1. Requiring prior knowledge for model specification, 2. Computationally expensive, 3. Sensitive to parameter estimation	Amayri et al. (2017)

Continued on next page

Table A.7 – Continued from previous page

Models (alphabetical order)	Main features and applications	Main category	Best-suited data type	Complexity	Strengths	Weaknesses	Selected references
Classification and Regression Trees (CART)	Decision tree for classification and regression	Supervised	Structured data with categorical and numerical features	Basic or Simple Models	1. Being interpretable, 2. Handling both categorical and numerical features, 3. Being non-parametric	1. Prone to overfitting, 2. Lack of robustness to small changes in data, 3. Can create complex trees	Chen et al. (2016), Koklu and Tutuncu (2019)
Convolutional Neural Network (CNN)	Deep learning method for (image) classification and (object) detection	Unsupervised	Image data	Advanced or complex Models	1. Capturing spatial patterns, 2. Translation-invariant, 3. Using hierarchical feature extraction	1. Requiring large amounts of labeled data, 2. Computationally expensive, 3. Lack of interpretability	Bao et al. (2021), Conti et al. (2014)
Decision Tree (DT)				Same as CART			
Density-Based Spatial Clustering of Applications with Noise (DBSCAN)	Clustering and outlier detection	Unsupervised	Structured data with spatial or density-based patterns	Basic or Simple Models	1. Discovering clusters of arbitrary shape, 2. Robust to noise and outliers, 3. Not requiring the number of clusters as input	1. Sensitive to the choice of parameters, 2. Difficulty in handling varying densities, 3. Not suitable for high-dimensional data	Novikova et al. (2020), Yan et al. (2016)
Extreme Gradient Boosting (XGBoost)	Ensemble learning method for classification and regression	Unsupervised	Structured data with numerical features	Intermediate Models	1. High predictive performance, 2. Handling missing data, 3. Effective feature selection	1. Requiring tuning of hyperparameters, 2. Longer training time compared to simple models, 3. Lack of interpretability	Lu et al. (2020b), Mohammadabadi et al. (2022)
Extreme learning model (XLM)	Machine Learning method for regression	Unsupervised	Structured data	Intermediate Models	1. Fast training and inference, 2. Good generalization performance, 3. Handling large-scale datasets efficiently	1. Limited interpretability, 2. Lack of fine-tuning options, 3. May require careful parameter tuning for optimal performance	Huang et al. (2006), Chen et al. (2016)
Gaussian mixture models (GMM)	Statistical methods for clustering and density estimation	Unsupervised	Structured data with probabilistic distributions	Intermediate Models	1. Capturing complex data distributions, 2. Flexible clustering, 3. Handling mixed data types	1. Sensitive to initialization, 2. Prone to local optima, 3. Computationally expensive	Xu et al. (2020)
Generative Adversarial Network (GAN)	Deep learning method for data generation and unsupervised learning	Unsupervised	Any type of data	Cutting-edge or Advanced Models	1. Generating realistic synthetic data, 2. Capturing complex data distributions	1. Training instability, 2. Mode collapse, 3. Challenging optimization	Chen and Jiang (2018), Chokwitthaya et al. (2019)

Continued on next page

Table A.7 – Continued from previous page

Models (alphabetical order)	Main features and applications	Main category	Best-suited data type	Complexity	Strengths	Weaknesses	Selected references
Hidden Markov Model (HMM)	Graphical-based models for sequential data modeling and prediction	Unsupervised	Sequential data with underlying hidden states	Intermediate Models	1. Capturing sequential dependencies, 2. Handling probabilistic modeling, 3. Handling missing data	1. Assuming a fixed number of hidden states, 2. Limited representation power, 3. Sensitive to initialization	Amayri et al. (2017), Candanedo et al. (2017b)
K-means	Clustering and data partitioning	Unsupervised	Structured data with numerical features	Basic or Simple Models	1. Simple and fast, 2. Scales well to large datasets, 3. Easy interpretation of cluster centroids	1. Requiring the number of clusters as input, 2. Sensitive to initialization, 3. Assuming spherical clusters	Pan et al. (2017), Prabhakaran et al. (2022)
k-Nearest Neighbors (k-NN)	Machine learning method for classification and regression	Unsupervised	Any type of data	Basic or Simple Models	1. Handling multi-class problems, 2. Suitable for large, dynamic datasets, 3. Handling noisy training data	1. Slower prediction for large datasets, 2. Sensitive to irrelevant features, 3. Memory-intensive	Soofi and Awan (2017)
Linear Discriminant Analysis (LDA)	Statistical methods for dimensionality reduction and classification	Supervised	Structured data with categorical features	Basic or Simple Models	1. Effective for multi-class problems of categorical inputs, 2. Handling multicollinearity, 3. Relying on interpretable linear transformations	1. Assuming normality and equal covariance matrices, 2. Limited representation power for complex data, 3. Prone to overfitting with small sample sizes	Chen et al. (2016), Neale et al. (2022)
Linear Regression	Statistical methods for regression and prediction	Supervised	Structured data with numerical features	Basic or Simple Models	1. Simple to comprehend and describe, 2. Guaranteeing the discovery of optimal weights, 3. Providing insights into feature importance and relationships	1. Incapable of handling non-linear relationships, 2. Sensitive to outliers, 3. Limited representation power for complex data patterns	Kim and Srebric (2017), Zheng et al. (2019)
Logistic Regression	Statistical methods for (binary) classification and probability estimation	Supervised	Structured data with numerical features	Basic or Simple Models	1. Interpretation of parameters is possible, 2. Handling categorical and numerical features and providing probabilistic outputs, 3. Regularization can be applied to prevent overfitting	1. Incapable of handling non-linear relationships, 2. Sensitive to outliers, 3. Limited representation power for complex data patterns	Stazi et al. (2017)

Continued on next page

Table A.7 – Continued from previous page

Models (alphabetical order)	Main features and applications	Main category	Best-suited data type	Complexity	Strengths	Weaknesses	Selected references
Long Short-Term Memory (LSTM)	Deep learning method for sequential data modeling and prediction	Semi-supervised	Sequential data with long-term dependencies	Advanced or Complex Models	1. Capturing long-term dependencies, 2. Handling variable-length sequences, 3. Mitigating the vanishing gradient problem	1. Expensive and prone to overfitting with small datasets, 2. Showing challenging interpretation of learned representations	Qolomany et al. (2017)
Naïve Bayesian Classifier (NBC)	Machine learning method for classification and probabilistic modeling	Semi-supervised	Structured data with categorical features	Basic or Simple Models	1. Fast training and prediction, 2. Handling high-dimensional data, 3. Providing probabilistic outputs	1. Assuming independence between features, 2. Limited representation power for complex dependencies, 3. Sensitive to irrelevant features	Fajilla et al. (2021), Fang and Hu (2014)
Principal Component analysis (PCA)	Statistical method for dimensionality reduction	Unsupervised	Structured data	Basic or Simple Models	1. Offering to reduce noise and redundancy in the data, 2. Improving computational efficiency and visualization, 3. Preserving the original structure and relationships inherent to the data	1. Assuming linear relationships between variables, which may limit its effectiveness in capturing complex nonlinear patterns, 2. Challenging to interpret with a large number of variables, 3. Sensitive to scaling and normalization of the data	Baird et al. (2017), Dridi et al. (2022)
Random Forest (RFo)	Ensemble learning method for classification and regression	Supervised	Structured data with numerical and categorical features	Intermediate Models	1. Handling high-dimensional data, 2. Robust to outliers, 3. Providing feature importance estimation	1. Less interpretable than decision trees, 2. Memory-intensive for large forests, 3. Slower inference compared to single decision trees	Parzinger et al. (2022), Wang et al. (2019a)
Recurrent Neural Network (RNN)	Deep learning method for sequential data modeling and prediction	Semi-supervised	Sequential data with dependencies	Intermediate Models	1. Capturing sequential dependencies, 2. Handling variable-length sequences, 3. Supporting online learning	1. Vulnerable to vanishing/exploding gradients, 2. Challenging long-term memory retention, 3. Computationally expensive	Chalapathy et al. (2021), Huchuk et al. (2019)
Rules-based methods	Classification and decision-making based on predefined rules	Unsupervised	Structured data with interpretable rules	Basic or Simple Models	1. Being Interpretable, 2. Providing fast inference, 3. Giving explicit rule-based decision-making	1. Limited representation power for complex data patterns, 2. Challenging in rule discovery, 3. Limited adaptability to unseen patterns	Aliero et al. (2022), Dorokhova et al. (2020)

Continued on next page

Table A.7 – *Continued from previous page*

Models (alphabetical order)	Main features and applications	Main category	Best-suited data type	Complexity	Strengths	Weaknesses	Selected references
Seasonal Autoregressive Integrated Moving Average (SARIMA)	Statistical method for time series forecasting	Specific	Time series data with seasonal patterns	Intermediate Models	1. Capturing and modeling seasonal patterns in time series, 2. Providing a flexible framework for incorporating autoregressive, moving average, and differencing components, 3. Handling non-stationary time series by differencing operations	1. Requiring determining the appropriate order of autoregressive (AR), integrated (I), and moving average (MA), 2. Assuming that the underlying data follows a linear relationship, 3. Struggling with irregular or highly volatile time series data	Fang and Lahdelma (2016), Ngo et al.
Seasonal Decomposition of Time Series (SDTL)	Statistical methods for time series decomposition and analysis	Specific	Time series data with seasonal components	Basic or Simple Models	1. Separating time series into trend, 2. Facilitating trend and seasonality analysis	1. Assuming additive or multiplicative decomposition, 2. Showing limited forecasting capabilities, 3. Being challenged with irregular patterns	Arief-Ang et al. (2018)
Shallow Neural Network (SNN)	Deep learning method for classification and regression	Unsupervised	Structured data with numerical features	Intermediate Models	1. Handling non-linear modeling, 2. Handling complex relationships, 3. Faster training compared to deeper architectures	1. Limited representation power for very complex data, 2. Vulnerable to overfitting, 3. Requiring careful hyperparameter tuning	Chalapathy et al. (2021), Manno et al. (2022)
Spectral Clustering	Machine learning method for unsupervised learning, particularly clustering	Unsupervised	Unlabeled data or data with no predefined class labels	Intermediate Models	1. Effective in clustering data points based on their similarity or affinity, even when the clusters have complex shapes, 2. Handling nonlinear relationships and capturing intricate structures in the data, 3. Less sensitivity to the shape and size of clusters vs traditional clustering algorithms	1. Requiring determining the number of clusters in advance, which can be challenging, 2. Computationally expensive, especially for large datasets, 3. Sensitivity to the choice of affinity or similarity measure	Ghaffar et al. (2022), Yang et al. (2017b)
Support Vector Machine (SVM)	Machine learning method for classification and regression	Supervised	Structured data with numerical features	Intermediate Models	1. Effective for high-dimensional data, 2. Robust to outliers, 3. Working well with small to medium-sized datasets	1. Computationally expensive for large datasets, 2. Sensitive to hyperparameters, 3. Limited interpretability of learned models	Kondratovich et al. (2013), Zuraimi et al. (2017)

Continued on next page

Table A.7 – Continued from previous page

Models (alphabetical order)	Main features and applications	Main category	Best-suited data type	Complexity	Strengths	Weaknesses	Selected references
Support Vector Regression (SVR)	Machine learning method for regression and prediction	Supervised	Structured data with numerical features	Intermediate Models	1. Effective for high-dimensional data, 2. Handling non-linear relationships, 3. Robust to outliers	1. Computationally expensive for large datasets, 2. Sensitive to hyperparameters, 3. Limited interpretability of learned models	Li and Dong (2018), Moradzadeh et al. (2020)
t-Distributed Stochastic Neighbor Embedding (t-SNE)	Statistical method for dimensionality reduction	Unsupervised	High-dimensional structured data	Intermediate Models	1. Capturing both local and global structure in the data, 2. Being effective in visualizing clusters, patterns, and relationships, 3. Revealing complex, nonlinear relationships	1. Being computationally expensive for large datasets, 2. Requiring careful parameter tuning for optimal results, 3. Showing outcomes sensitive to the initial configuration and random seed	Kim and Cho (2021)
Uniform Manifold Approximation and Projection (UMAP)	Statistical method for (non-linear) dimensionality reduction	Unsupervised (or even semi-supervised)	Any type of data	Advanced or Complex Models	1. Capturing non-linear relationships preserving complex structures within the data, 2. Highly scalable and handling large datasets efficiently, 3. Flexibility in terms of parameter tuning and can be adapted to various types of data and applications	1. Sensitive to the choice of hyperparameters (number of neighbors, minimum distance), 2. Lack of interpretability, 3. May struggle to handle outliers or anomalies in the data and may affect the overall representation of the data	Khan et al. (2023)
Variational Auto-encoder (VAE)	Deep learning method for data generation and dimensionality reduction	Semi-supervised or unsupervised	Any type of data	Advanced or Complex Models	1. Capturing latent representations, 2. Generating synthetic data, 3. Handling high-dimensional data	1. Being challenged by training dynamics, 2. Showing mode collapse, 3. Less interpretability compared to traditional auto-encoders	Konstantakopoulos et al. (2019), Mendes et al. (2023)

A.8 Current problems with the application of machine learning algorithms for occupancy activities and event detection

Table A.8 – Some problems associated with the use of machine learning for classification and regression and examples of methods most and least sensitive to the identified weaknesses.

Limitations	Explanation and impact	Most sensitive methods	Less sensitive or able to mitigate limitation
Data scarcity	The lack of data can be considered as one of the most penalizing limitations, as an insufficient number of learning examples can compromise the ability of models to generalize correctly.	e.g., ANN (it typically requires a large labeled dataset to effectively learn complex patterns and representations. Training ANN from scratch with limited data can lead to poor generalization performance).	e.g., Naïve Bayesian Classifier (it relies on simplifying assumptions of independence between variables and still provides reasonable predictions by leveraging the information available in the training data).
Overfitting	Overlearning, or overfitting, can lead to poor performance on new data, as the model has learned the specific details of the training set too well without generalizing properly. All learning methods may be subject to overfitting if the models are too complex for the amount and quality of training data available.	e.g., complex models such as Deep NN (their high capacity and flexibility make them prone to overfitting, however techniques such as regularization, dropout, early stopping, or reducing the number of layers or neurons can help mitigate overfitting).	Regularized regression methods, such as Lasso regression used for Bayesian Networks (they control complexity of model and reduce tendency to overfit the training data).
Imbalanced data	When the classes of events are unbalanced, i.e., there are many more examples of one class than another, this can lead to a bias in the performance of the learning method, favoring the majority class.	e.g., SVM (it aims to find an optimal hyperplane to separate different classes, however, when one class significantly outnumbers the others, it tends to prioritize the majority class and may struggle to accurately classify the minority class).	e.g., Synthetic Minority Oversampling Technique - SMOTE (oversampling technique that addresses the class imbalance problem by generating synthetic examples of the minority class).

Continued on next page

Table A.8 – *Continued from previous page*

Limitations	Explanation and impact	Most sensitive methods	Less sensitive or able to mitigate limitation
Non-stationarity	Real environments are often dynamic and subject to temporal changes. When the characteristics of the data change over time, the models may have difficulty adapting to these variations, which can lead to a decline in performance.	Time-dependent models (e.g., ARIMA-SARIMA) (they assume that the underlying time series data exhibit stationarity, however, many real-world time series exhibit non-stationary behavior, characterized by trends, seasonality, or changing statistical properties).	e.g., RNN, such as LSTM (it captures temporal dependencies and patterns in the data, making them more robust to changes over time).
Sensitivity to parameters	Learning methods often require parameter settings to optimize their performance. However, poor parameter selection can lead to sub-optimal results or even instability in predictions.	e.g., SVM (it relies on tuning hyperparameters, such as the regularization parameter (C) and the kernel parameters, to find the optimal decision boundary and whose choice significantly influence its performance and generalization).	e.g., Random Forest (the algorithm aggregates the predictions of multiple individual trees, reducing the impact of parameter variations in each tree).
Interpretability	Difficulty in interpreting models can be an important limitation, especially if clear and understandable explanations are needed to make decisions or communicate results. Understanding how and why these models make decisions can be difficult, which may limit their acceptance and use in some sensitive applications.	e.g., CNN (it is highly complex due to the lack of transparency in their internal workings. The intricate layers and connections make it challenging to interpret how the input data is transformed and how decisions are made).	e.g., Decision trees (their hierarchical structure allows for clear visualization and understanding of the decision-making process).
Sensitivity to noise	Real data may contain noise or errors, which can affect the performance of learning methods. Models may be sensitive to outliers or measurement errors, which can lead to incorrect predictions.	e.g., k-NN (it is a non-parametric algorithm that classifies or predicts based on the similarity of input data points to their neighbors, thus, noisy data points that deviate significantly from the underlying patterns can lead to incorrect classifications or predictions).	e.g. SVM (by adjusting its regularization parameter, it can be tuned to be more or less sensitive to individual data points, including noisy ones).

Continued on next page

Table A.8 – *Continued from previous page*

Limitations	Explanation and impact	Most sensitive methods	Less sensitive or able to mitigate limitation
Need for labeled data	Many learning methods require annotated data, i.e., data labelled by human experts. Manual annotation of data can be costly and time-consuming, limiting the availability of large, annotated data sets.	e.g., logistic Regression methods (they rely on modeling the relationship between input features and a binary outcome and it requires a sufficient amount of labeled data).	e.g., GAN (the generator part learns to capture the underlying data distribution without relying on explicit labels or annotations and it is possible to generate synthetic labeled data that can be used to augment the limited annotated data).
Missing data	Missing data can lead to problems in learning, as important information can be lost, requiring imputation strategies or missing data management techniques.	e.g., GMM (it does not explicitly model missing data or provide mechanisms for imputation and may struggle to accurately estimate the parameters and capture the underlying patterns in the data when faced with missing data).	e.g., Bayesian Networks (they encode prior knowledge and conditional dependencies between variables using probability distributions and thus can estimate missing values by considering the observed variables and their dependencies).

ANNEX ON CHAPTER 5

B.1 Review of self-learning supervised method

The methods for self-supervised learning can be primarily categorized into three types, 1. Context based, 2. Temporal based, 3. Contrastive based

Self-supervised learning leverages the inherent context within the data to construct pretext tasks. For instance, in the Natural language processing (NLP) domain, one of the most important algorithms is Word2Vec (Mikolov et al.2015). This model mainly predicts a central word based on surrounding words or predicts surrounding words based on a central word within a sentence. In the domain of image processing, Carl et al.2015 used a technique known as "Jigsaw" to design auxiliary tasks. In this approach, an image is divided into nine segments, and the model is trained to predict the relative positions of these segments, enhancing its semantic understanding. Subsequent works (Norozzi et al.2016) have expanded this Jigsaw concept, introducing more complex or challenging tasks. For example, an image is still divided into nine pieces but shuffled in one of 64 predefined orders. The model then learns to classify the shuffled sequence, requiring it to grasp more intricate relative positional information. The insight gained from this work is that stronger supervision, or harder auxiliary tasks, leads to better performance.

Deepak et al.2016 also used the inpainting technique, where a portion of an image is randomly removed, and the model learns to predict the missing part based on the remaining image. Only when the model truly understands the meaning represented by this graph can it effectively complete it. This technique parallels the MASK LM training approach used in BERT (Devlin et al.2019) for NLP, where random words are removed from a sentence, and the model learns to predict them.

Zhang et al.2016 proposed a colorization tasks for the self-supervised learning. In this case, a model is trained to predict the colors in a grayscale image. The model must understand the semantic information in the image to color it appropriately, like making the sky blue and the grass green. These color-prediction generative models have opened

up new avenues in research, showing that any decoupled features can be used for mutual supervision. The renowned Split-Brain Autoencoders (Zhang et al.2017) work on a similar principle. They first divide the raw data into two parts and use information from one part to predict the other, finally synthesizing the complete data. Unlike traditional encoders, this prediction mechanism necessitates that the model genuinely understands the semantic information in the data. Hence, it indirectly constrains the encoder to train beyond just pixel-wise levels and to consider more semantic features. Next, we discuss self-supervised contexts found through data augmentation techniques. Gidaris et al.2018 proposed a study which involved taking an input image and rotating it at various angles; the model aims to predict the degree of rotation which can be seen in Figure B.1. This straightforward idea led to significant gains, highlighting the benefits of data augmentation in self-supervised learning.

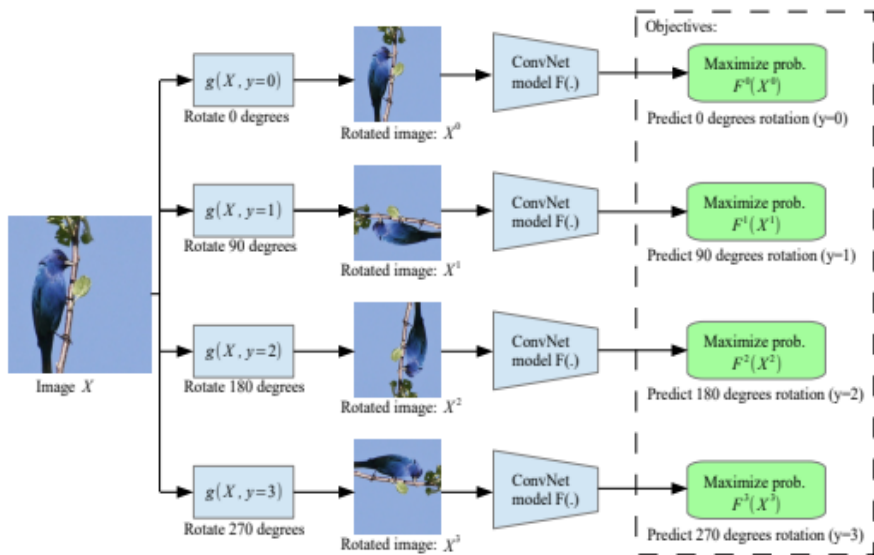


Figure B.1 – Figure rotation pretext

Most previously discussed methods focus on the sample itself, such as rotation, color, and cropping. However, constraints also exist between samples. Here, we introduce self-supervised learning methods that leverage temporal constraints. Video data best exemplifies temporality. One approach is frame-based similarity (Sermanet et al.2017), where adjacent frames in a video are assumed to have similar features while those far apart do not. The model is trained with such similar (positive) and dissimilar (negative) samples. Another idea is unsupervised tracking (Wang et al.2015). In this method, object tracking

frames are obtained from a large number of unlabeled videos. The features of an object in different frames should be similar (positive), while those of different objects should be dissimilar (negative). Beyond feature similarity, the sequential order in videos also serves as a self-supervised cue. For instance, Misra et al.2016 proposed a method based on sequential constraints. It uses both correct and incorrect video sequences as positive and negative samples for training. Essentially, a model is designed to determine whether a given video sequence is in the correct order. This sequence-based constraint has also been applied to dialogue systems. Wu et al.2019 proposed a self-supervised dialogue learning model based on this concept. The aim is to improve the coherence of generated dialogues, ensuring that machine-generated responses align with prior conversational style and habits. The model is trained to predict the correct sequence from a large corpus of historical data, generating more coherent dialogues through adversarial training upon completion.

The third category of self-supervised learning methods is based on contrastive constraints, which build representations by learning to encode the similarity or dissimilarity between two entities. Hjelm et al.2019 proposed a DIM algorithm. The core idea behind DIM is to differentiate between global features (the final output of the encoder) and local features (features from intermediate layers of the encoder). The model is trained to determine whether the global and local features originate from the same image. Here, the positive sample is the local feature from the same image, and the negative sample is the local feature from a different image. This pioneering work has been adapted for other domains, such as deep graphs (Velickovic et al.2018). Contrastive predictive coding (CPC) is another contrastive-constraint-based self-supervised framework that can be applied to any data form that can be represented in a sequential manner, such as text, speech, video, or even images (which can be viewed as sequences of pixels or blocks). CPC primarily employs an autoregressive approach to encode shared information between data points separated by multiple time steps, where the positive sample is the input at a future time t , and the negative sample is randomly sampled from other sequences. The main idea behind CPC is to predict future data based on past information, and it's trained through sampling.

B.2 Estimation BN architecture

Before introducing the four groups of BN structure (score-based, constraints based, hybrid, and domain knowledge), we have to introduce the Bayesian Information Criterion

(BIC) which is commonly used as a scoring function. BIC, also known as Schwarz criterion, is an approximation of the Bayesian marginal likelihood of the data given the model. BIC balances the goodness of fit of the model with its complexity by introducing a penalty term proportional to the number of parameters in the model.

The BIC score for a BN is defined as follows:

$$BIC(G) = \log P(D|G) - \frac{d}{2} \log N \quad (\text{B.1})$$

where:

- G is the graph (network structure)
- D is the data
- $P(D|G)$ is the likelihood of the data given the graph
- d is the number of independent parameters in the model
- N is the number of data points (sample size)

The likelihood term, $\log P(D|G)$, measures the goodness of fit of the model to the data, while the penalty term, $\frac{d}{2} \log N$, discourages overly complex models. The objective is to find a network structure that maximizes the BIC score.

B.2.1 Score based structure estimation

For the Score-based structure estimation, Hill Climbing starts with an initial network structure and iteratively refines the structure by performing local operations, such as adding, deleting, or reversing edges, to optimize a scoring function. The algorithm terminates when no further improvement can be made or a predefined stopping criterion is met. The limitation of the hill climbing method is that it can be stuck in the optima, and it can be computationally expensive, especially when the search space is large.

The Hill Climbing algorithm for BN structure estimation can be described as follows:

1. Initialize a network structure, G_0 .
2. Repeat until a stopping criterion (The algorithm stops when no modification leads to a better BIC score than the current network) is met:
 - (a) For each possible modification (add, delete, or reverse an edge) of the current network structure, G_t , compute the BIC score of the resulting network, G_{t+1} .
 - (b) Choose the modification that results in the highest BIC score.

-
- (c) If the highest BIC score of the modified networks is greater than the BIC score of the current network, G_t , update the current network to the modified network with the highest BIC score, G_{t+1} .
 - (d) Otherwise, terminate the algorithm.

Tabu search is another Score-based structure estimation that can be used for estimating the structure of BNs. It is a global search method that aims to overcome the limitations of local search algorithms, such as Hill Climbing, which can get stuck in local optima. Tabu search employs a memory structure called a tabu list to guide the search process and prevent revisiting recently explored solutions.

The Tabu search algorithm for BN structure estimation can be described as follows:

1. Initialize a network structure, G_0 , and an empty tabu list, T .
2. Set the best solution found so far, G_{best} , to the initial network structure, G_0 .
3. Repeat until a stopping criterion (The algorithm stops if there has been no improvement in the best solution found so far, G_{best} , for a predefined number of iterations) is met:
 - (a) Generate a set of candidate network structures, C , by applying all possible single modifications (add, delete, or reverse an edge) to the current network structure, G_t , subject to the constraint that no modification results in a network structure in the tabu list, T .
 - (b) Choose the candidate network structure, G_{t+1} , with the highest score according to a scoring function (e.g., BIC) from the set C .
 - (c) If the score of G_{t+1} is higher than the score of G_{best} , update G_{best} to G_{t+1} .
 - (d) Update the tabu list, T , by adding the reverse modification applied to obtain G_{t+1} from G_t , and removing the oldest modification if the size of T exceeds a predefined maximum size, T_{max} .
 - (e) Set the current network structure, G_t , to the chosen candidate network structure, G_{t+1} .

The Chow-Liu algorithm is another Score-based structure estimation for estimating the structure of a BN that uses a tree-structured network to maximize the mutual information between variables. The Chow-Liu algorithm aims to find the best tree-structured BN that approximates the joint distribution of the data. This is achieved by minimizing the Kullback-Leibler (KL) divergence between the true joint distribution and the one represented by the tree-structured BN.

The Chow-Liu algorithm can be described as following steps:

1. Calculate the mutual information $I(X_i; X_j)$ for all pairs of variables (X_i, X_j) in the dataset.
2. Create a complete graph with the variables as nodes and the mutual information between pairs of variables as edge weights.
3. Find the maximum spanning tree (MST) of the complete graph using an algorithm such as Kruskal's or Prim's algorithm. This tree will represent the optimal tree-structured BN.

For the mutual information, the mutual information between two variables X_i and X_j is a measure of the dependency between the variables and is defined as follows:

$$I(X_i; X_j) = \sum_{x_i \in \mathcal{X}_i} \sum_{x_j \in \mathcal{X}_j} P(x_i, x_j) \log \frac{P(x_i, x_j)}{P(x_i)P(x_j)} \quad (\text{B.2})$$

where \mathcal{X}_i and \mathcal{X}_j are the domains of the variables X_i and X_j , respectively, and $P(x_i, x_j)$ is the joint probability distribution of X_i and X_j . The mutual information is non-negative and symmetric, i.e., $I(X_i; X_j) = I(X_j; X_i)$.

A spanning tree of an undirected graph is a subgraph that includes all the vertices and is a tree. The MST is a spanning tree with the maximum possible sum of edge weights. The MST can be found using algorithms such as Kruskal's or Prim's algorithm. The MST of the complete graph created in the Chow-Liu algorithm represents the optimal tree-structured BN.

Score-based structure estimation of the BN is a method that relies on a scoring function to evaluate how well the network structure fits the data. These algorithms have some advantages, such as handle large and complex data sets, as it does not require conditional independence tests for each variable pair, and it can avoid ambiguity, as it can select a unique network structure that has the highest score among the equivalent ones. Also, It can incorporate prior knowledge, such as Bayesian priors, to influence the scoring function and the search algorithm. But, they also have some drawbacks, such as it can be biased by the choice of the scoring function, and it can be trapped in local optima, as it depends on the quality of the search algorithm and the initial network structure.

B.2.2 Constraint-based structure estimation

Constraint-based structure learning is a method for learning the structure of BNs from data. The key idea behind this approach is to identify conditional independence relationships in the data and construct a directed acyclic graph (DAG) based on these relationships. Because BN is a probabilistic graphical model that represents a set of variables and their conditional dependencies via a directed acyclic graph (DAG). Let X_1, X_2, \dots, X_n be a set of random variables. A BN B is a pair (G, P) , where G is a DAG with nodes corresponding to the variables X_1, X_2, \dots, X_n , and P is a set of conditional probability distributions associated with each variable given its parents in the graph. Then, Constraint-based structure learning starts by identifying (conditional) independence relationships in the data using hypothesis tests. For this purpose, we use the chi-squared test for conditional independence. The test statistic is defined as:

$$\chi^2 = N \sum_{i,j,k} \frac{(O_{ijk} - E_{ijk})^2}{E_{ijk}} \quad (\text{B.3})$$

where N is the number of samples, O_{ijk} is the observed frequency of the combination of variable values i, j , and k , and E_{ijk} is the expected frequency under the null hypothesis of independence.

The Peter-Clark (PC) algorithm is the method that we tried in here, PC algorithm is a popular constraint-based structure learning algorithm that proceeds in two phases: skeleton identification and orientation of edges. The algorithm can be summarized as follows:

1. Initialize the undirected graph with nodes corresponding to variables and fully connected edges.
2. For each pair of variables X_i and X_j and increasing conditioning set size $k = 0, 1, \dots, K$:
 - (a) If there exists a conditioning set Z of size k such that $X_i \perp X_j \mid Z$, remove the edge between X_i and X_j .
3. For each pair of non-adjacent variables X_i and X_j :
 - (a) If there exists an intermediate variable X_k such that $X_i \rightarrow X_k \rightarrow X_j$, orient the undirected edges as $X_i \rightarrow X_k$ and $X_k \rightarrow X_j$.
4. Return the resulting DAG.

constraint-based algorithms are based on conditional independence tests to infer the network structure from the data. These algorithms have some advantages, such as capturing the causal relationships among the variables, handling missing values and continuous variables, and incorporating prior knowledge. However, they also have some drawbacks, such as being sensitive to errors in the conditional independence tests, being computationally expensive, and being ambiguous.

B.2.3 Hybrid structure estimation

Max-Min Hill-Climbing (MMHC) is a hybrid algorithm for learning the structure of BNs. It combines the strengths of constraint-based and score-based approaches. MMHC consists of two main phases: the first phase is a constraint-based phase, where the algorithm identifies a skeleton of the network by applying the max-min parents and children (MMPC) algorithm; the second phase is a score-based phase, where the algorithm refines the structure using a greedy hill-climbing algorithm. The MMHC algorithm is known for its ability to efficiently learn the structure of BNs, even when the number of variables is large.

The MMPC algorithm aims to find the Markov blanket of each variable, which includes its parents, children, and children's other parents. The algorithm operates in two stages, a forward search and a backward elimination. Given a variable X , the algorithm starts with an empty set of candidate parents and children, denoted as $PC(X)$.

1. **Forward search:** In this stage, the algorithm iteratively adds variables to the $PC(X)$ set based on the conditional mutual information $I(X;Y|Z)$, where Y is a candidate variable and Z is a subset of already included variables in $PC(X)$. The algorithm stops adding variables when no further significant dependencies are detected.
2. **Backward elimination:** In the second stage, the algorithm tests the variables already included in $PC(X)$ for conditional independence, given the other variables in $PC(X)$. If a variable is found to be conditionally independent, it is removed from the set. This process continues until all remaining variables in $PC(X)$ are conditionally dependent on X given the others.

After obtaining the skeleton of the network using the MMPC algorithm, the MMHC algorithm refines the structure using a greedy hill-climbing search. This search aims to maximize the score of the BN structure. The BN structure score is given by:

$$S(G) = \sum_{i=1}^n S_i(\text{Pa}_i(G)), \quad (\text{B.4})$$

where G represents the network structure, n is the number of variables, $\text{Pa}_i(G)$ is the set of parents of variable i in the structure G , and S_i is the score of variable i given its parent set.

The hill-climbing search starts with an initial structure and iteratively modifies it by applying local operations, such as adding, deleting, or reversing an edge. The search continues until no operation can improve the structure score or a predefined stopping criterion is reached. The following local search operations are considered in MMHC:

1. **Addition:** Adding an edge between two variables if it is not already present and does not introduce a cycle.
2. **Deletion:** Deleting an existing edge between two variables.
3. **Reversal:** Reversing the direction of an existing edge, if the new edge direction does not introduce a cycle.

The MMHC algorithm can be summarized as follows:

1. For each variable X , apply the MMPC algorithm to find its Markov blanket $\text{MB}(X)$.
2. Construct an initial network structure by connecting variables in each Markov blanket, avoiding cycles.
3. Refine the initial network structure using a hill-climbing search to maximize the BN structure score.

MMHC is a popular algorithm for learning BN structures due to its efficiency and scalability. It combines the advantages of constraint-based and score-based methods, providing a powerful approach for structure learning in complex domains. Such as it can reduce the computational cost, as it uses conditional independence tests only for the Markov blanket of each variable, rather than for all variable pairs. And, it can improve the accuracy, as it uses both conditional independence tests and scoring functions to infer the network structure. It also can avoid local optima, as it uses a tabu list to prevent revisiting previous network structures.

B.2.4 Domain knowledge

Domain knowledge in BN structure estimation refers to the use of expert knowledge or information about the relationships between variables to guide the construction of the network, so we have to understand of the causal or influential relationships among variables. Firstly, we need list all the variables that are relevant to the problem which we are trying to model, these variables will be the nodes in the BN. Secondly, we need determine the scope of the model, the scope will guide you in establishing the relationships among variables. Based on your domain knowledge, identify which variables have direct causal relationships or strong correlations. For example, window status will likely influence indoor temperature. Then, we need create an initial network structure by drawing arrows from "parent nodes to "child nodes", the terms "parent nodes" and "child nodes" are used to describe the relationships between variables. Specifically, a parent node is a variable that directly influences another variable, and the influenced variable is called the "child node." In terms of conditional probabilities, the parent nodes are the conditions given which the child node's probability is defined. In the graphical representation of a BN, an arrow points from the parent node to the child node to signify this causal or influential relationship. Once the network is built with these relationships in place, we can perform probabilistic inference to find the conditional probabilities of interest.

B.3 CPD parameters learning methods

B.3.1 Maximum Likelihood Estimation (MLE)

MLE is a frequentist approach to estimate the parameters of a model. The main idea behind MLE is to find the parameter values that maximize the likelihood function. The likelihood function measures how likely it is to observe the given data under the chosen parameter values. Mathematically, given a set of observed data $D = \{x_1, x_2, \dots, x_n\}$ and a parameterized model with parameters θ , the likelihood function is defined as:

$$L(\theta | D) = P(D | \theta) \propto \prod_{x_i \in D} \theta_{x_i | pa(x_i)} \quad (\text{B.5})$$

The Maximum Likelihood Estimate $\hat{\theta}_{MLE}$ is the value of θ that maximizes the likelihood function:

$$\hat{\theta}_{MLE} = \arg \max_{\theta} L(\theta|D) \quad (\text{B.6})$$

MLE has the advantages such as being relatively simple to compute and often having good asymptotic properties. It does not require prior knowledge about the parameter values, but it is also sensitive to the choice of the likelihood function. It can sometimes produce biased estimates for small sample sizes. It does not provide a full probability distribution over the parameter values; it only gives point estimates.

B.3.2 Bayesian Estimation

Bayesian Estimation is a probabilistic approach to estimate the parameters of a model. In contrast to MLE, Bayesian Estimation incorporates prior knowledge or belief about the parameter values in the form of a prior distribution. The prior distribution is combined with the likelihood function using Bayes' theorem to compute the posterior distribution over the parameter values:

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)} \quad (\text{B.7})$$

where $P(\theta|D)$ represent the posterior distribution of the parameters θ , given the observed data D , $P(D|\theta)$ describes, as previously written, the likelihood function describing the probability of observing the data D , given the parameters θ , $P(\theta)$ is the prior distribution representing our initial beliefs about the parameters θ .

$P(D)$ is the marginal likelihood or evidence, which is the probability of observing the data D regardless of the parameters θ . It serves as a normalization constant to ensure that the posterior distribution integrates to 1. The posterior distribution represents the updated beliefs about the parameter values after observing the data. The likelihood function $P(D|\theta)$ is modeled using the Multinomial distribution. The prior distribution $P(\theta)$ is modeled using the Dirichlet distribution. In our case study, even though window status and occupancy are binary (simulated case), their relationships with other variables like 'Electricity', 'Cooling', 'Solar', etc. in the Bayesian network (which have multiple states) lead the choice of a Multinomial distribution, especially when considering joint or conditional probabilities. As the Dirichlet distribution serves as the conjugate prior for the Multinomial distribution, simplifying the Bayesian updating process.

Bayesian estimation can provide a full probability distribution over the parameter values, allowing for uncertainty quantification. Additionally, it allows incorporating prior

knowledge or belief about the parameter values, which can lead to more robust estimates in cases where MLE might be sensitive to the choice of the likelihood function or when dealing with small sample sizes. However, Bayesian Estimation can be computationally expensive, particularly for high-dimensional models or complex likelihood functions. And the choice of prior distribution can have a significant impact on the posterior distribution, which might be undesirable in some situations. In some cases, the choice of an appropriate prior might be subjective or unclear, leading to potential biases in the estimation.

If we set the amount of prior information, expressed as the effective or “equivalent” sample size (ESS), to a value, say M , we can interpret this as if we have seen M virtual observations from the prior distribution that conform perfectly to our prior beliefs. In the case of the Dirichlet-Multinomial conjugate prior, the ESS prior is the sum of the $\alpha_{I|s,w}$ parameters for each combination of ‘S’ and ‘W’. For example, the CPD of ‘I’ (Indoor temperature) given ‘S’ (Solar gain) and ‘W’ (windows), the multinomial likelihood function for the observed data is given by:

$$L(\theta_{I|S,W} | X) = \prod_{s=1}^S \prod_{w=1}^W \prod_{I=1}^C \theta_{I|s,w}^{n_{I|s,w}} \quad (\text{B.8})$$

Where S , W , and I are the number of discrete bins for solar gain, windows, and indoor temperature, respectively. $n_{I|s,w}$ is the observed count of indoor temperature = I when solar gain = s and windows = w.

The Dirichlet distribution is the conjugate prior for the multinomial distribution and is given by:

$$P(\theta_{I|S,W} | \alpha) = \frac{1}{Z(\alpha)} \prod_{s=1}^S \prod_{w=1}^W \prod_{I=1}^C \theta_{I|s,w}^{\alpha_{I|s,w}-1} \quad (\text{B.9})$$

Where $\alpha_{I|s,w}$ are the parameters of the Dirichlet distribution, and $Z(\alpha)$ is a normalization constant.

Using Bayes’ theorem, the posterior distribution of $\theta_{I|S,W}$ is also a Dirichlet distribution:

$$P(\theta_{I|S,W} | X, \alpha) \propto \prod_{s=1}^S \prod_{w=1}^W \prod_{I=1}^C \theta_{I|s,w}^{n_{I|s,w} + \alpha_{I|s,w} - 1} \quad (\text{B.10})$$

By increasing the ESS prior value, we are giving more weight to your prior beliefs, and the updated estimates of $\theta_{I|S,W}$ will be pulled closer to the prior beliefs.

When we set the ESS prior to a large value, we inform the Bayesian estimation process that we hold strong prior beliefs, and that these beliefs should carry more weight when

estimating the CPDs' parameters. In other words, the estimation process will rely more on the prior information and less on the actual data when updating the CPDs' parameters.

In the simulation study, the dataset was judiciously split into training and test sets. Specifically, 75% was allocated for training, while the remaining 25% was designated for testing. In our Bayesian estimation framework, we employed the "BDeu" (Bayesian Dirichlet equivalent uniform) as our prior type. This choice represents a non-informative prior, suggesting that before any data observation, all potential outcomes or states of our variables are perceived with equal likelihood. The equivalent sample size (ESS) of this prior is set at 10. Practically, this can be conceived as introducing 10 pseudo-observations that perfectly align with a uniform belief. The likelihood distribution is formulated based on the observed data. The structural design of our Bayesian network specifies the inter-variable relationships, and we estimate the conditional probability distributions (CPDs). These CPDs encapsulate the likelihoods. For instance, the likelihood of registering a specific "Occupancy" state, given the states of other variables, is inherently captured in its CPD.

The Figure B.2 shows the prior distribution of the variable occupancy when employing the "BDeu" (Bayesian Dirichlet equivalent uniform) prior with an ESS of 10.

The Beta distribution, characterized by parameters $\alpha = \beta = 5$ (half of the ESS), exhibits symmetry around 0.5. This implies that, in the absence of any observed data, the assumption is that both states (e.g., Occupancy = 0 or 1) are equally probable. This is evident from the distribution's peak at 0.5. The peak's density value is influenced by the shape parameters α and β .

This visualization elucidates that prior to any data observation, the inherent belief is an equal likelihood of either state (e.g., the room being occupied or vacant). This represents a non-informative or weakly informative prior, indicating an absence of strong prior beliefs about the probability of either state. As data is acquired and beliefs are updated, the posterior distribution may undergo shifts based on the observed data. However, the prior commences from this neutral stance.

B.4 Application of the BN to the real case study

B.4.1 Bayesian network architecture for real case

The BN network for the simulation case study was provided in Chapter 5. In this annex, it is shown for the real case study. The variables for the real case study as below,

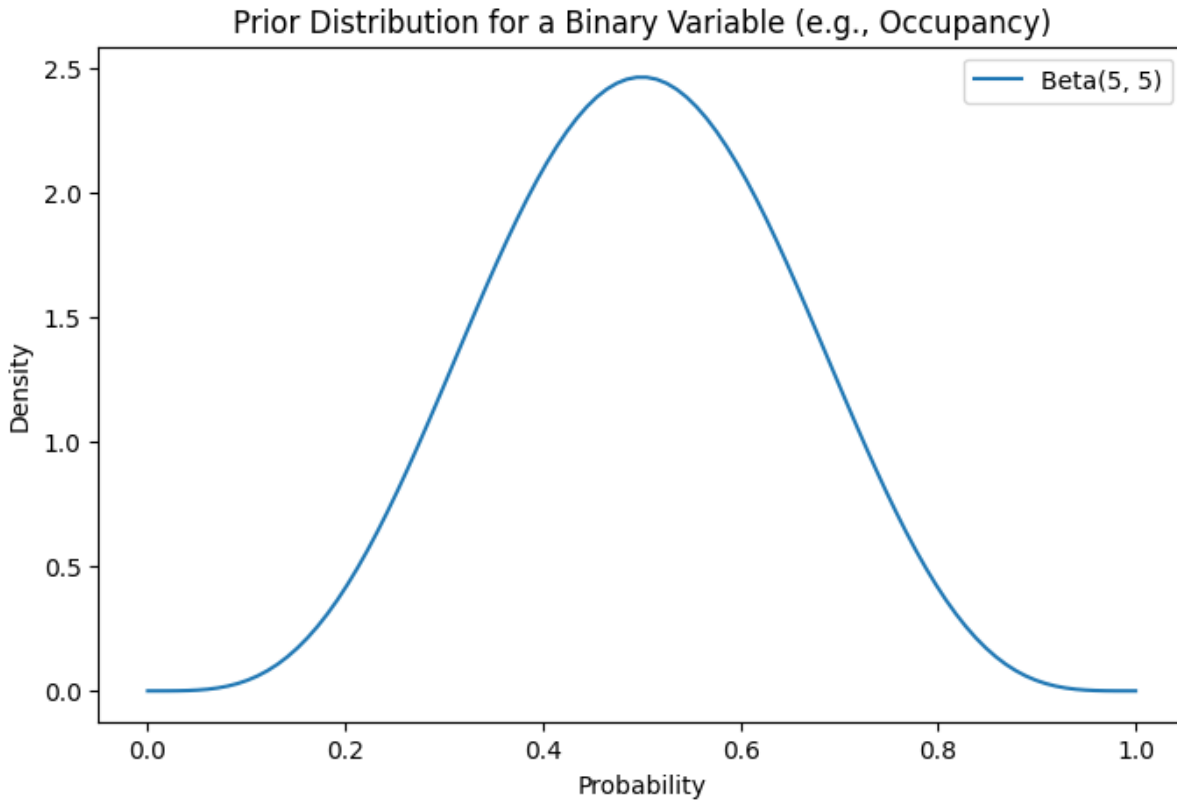


Figure B.2 – prior distribution of a variable

- W: Window status (0, 1, 2, 4)
- c: CO₂ level
- O: TVOC level
- S: Temperature
- H: Humidity
- I: Light intensity
- Ou: Sound level
- Oi: Outdoor weather

Based on the domain knowledge, BN structure for real data can be given as follow,
 The real case BN can be written as FigureB.3

B.4.2 Joint Probability Distribution in BNs in real case

Using the real case study BN structure, we can compute the joint probability of a specific assignment, say $P(W = w, c = c, O = o, S = s, H = h, I = i, Ou = u, Oi = l)$.

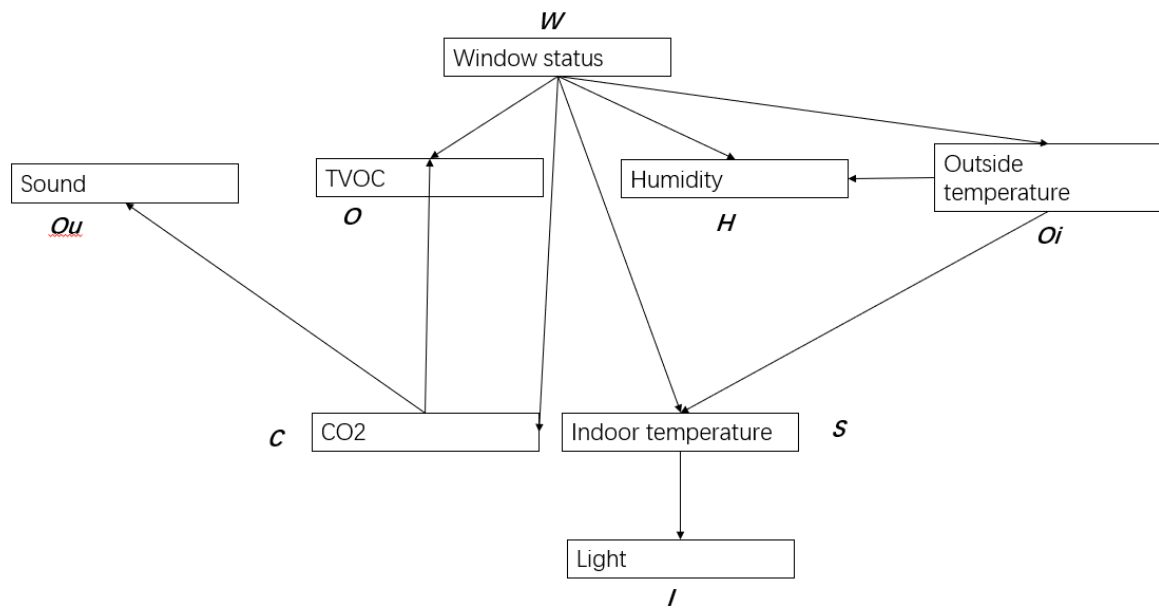


Figure B.3 – BN structure of the real case study

Following the steps outlined above:

1. Assign values: $W=w$, $c=c$, $O=o$, $S=s$, $H=h$, $I=i$, $O_u=u$, $O_i=l$
2. Look up the corresponding conditional probabilities for each node given the values of its parents:
 - $P(W = w)$
 - $P(c = c|W = w)$
 - $P(O = o|W = w, c = c)$
 - $P(S = s|W = w, O_i = l)$
 - $P(H = h|W = w, O_i = l)$
 - $P(I = i|S = s)$
 - $P(O_u = u|c = c)$
 - $P(O_i = l|W = w)$
3. Multiply the conditional probabilities obtained in step 2:

$$\begin{aligned}
& P(W = w, c = c, O = o, S = s, H = h, I = i, Ou = u, Oi = l) \\
&= P(W = w) \cdot P(c = c|W = w) \cdot P(O = o|W = w, c = c) \\
&\quad \cdot P(S = s|W = w, Oi = l) \cdot P(H = h|W = w, Oi = l) \\
&\quad \quad \cdot P(I = i|S = s) \cdot P(Ou = u|c = c) \\
&\quad \quad \quad \cdot P(Oi = l|W = w)
\end{aligned} \tag{B.11}$$

The result of this computation is the joint probability of the specific assignment of values $W = w, c = c, O = o, S = s, H = h, I = i, Ou = u, Oi = l$ in the BN. By following the steps outlined above, we can compute the joint probability of any specific assignment of values to the variables in the network. This joint probability distribution forms the basis for performing inference tasks in BNs, such as computing conditional probabilities and making predictions based on observed evidence.

B.4.3 Variable elimination real case

Using the real case BN as an example, the network structure is recalled in here: With the discretized data of each variables as follows:

- Windows (W): 0, 1, 2, 3,4
- CO₂ (c): A, B, C, D
- TVOC (O): A, B, C, D
- Temperature (S): A, B, C, D
- Humidity (H): A, B, C, D
- Light (I): A, B, C, D
- Sound (Ou): A, B, C, D
- Weather (Oi): A, B, C, D

Suppose we want to compute the probability distribution of the window status (W) given the following evidence:

$$c = A, \quad O = B, \quad S = C, \quad H = D, \quad I = A, \quad Ou = B, \quad Oi = C \tag{B.12}$$

We can use the Variable Elimination algorithm to compute the conditional probability distribution $P(W|c = A, O = B, S = C, H = D, I = A, Ou = B, Oi = C)$. In this case, our query variable is W, and the evidence variables are c, O, S, H, I, Ou, and Oi.

For the elimination order, we can choose the following ordering: Ou, I, H, S, O, c, Oi.

Now, we will eliminate each variable according to this order:

- Eliminate Ou: Compute the factor $\phi_{Ou}(c) = \sum_{Ou} P(Ou|c)$, and sum out Ou.
- Eliminate I: Compute the factor $\phi_I(S) = \sum_I P(I|S)$, and sum out I.
- Eliminate H: Compute the factor $\phi_H(W, Oi) = \sum_H P(H|W, Oi)$, and sum out H.
- Eliminate S: Compute the factor $\phi_S(W, Oi, c) = \sum_S P(S|W, Oi)P(c|S)$, and sum out S.
- Eliminate O: Compute the factor $\phi_O(c) = \sum_O P(O|c)$, and sum out O.
- Eliminate c: Compute the factor $\phi_c(W, Oi) = \sum_c P(c|W)\phi_O(c)\phi_{Ou}(c)$, and sum out c.
- Eliminate Oi: Compute the factor $\phi_{Oi}(W) = \sum_{Oi} P(Oi|W)\phi_H(W, Oi)\phi_S(W, Oi, c)$, and sum out Oi.

Now, we have obtained the factor $\phi_{Oi}(W)$, which represents the unnormalized probability distribution of the window status (W) given the evidence. To normalize the distribution, we can compute the following:

$$P(W|c = A, O = B, S = C, H = D, I = A, Ou = B, Oi = C) = \frac{\phi_{Oi}(W)}{\sum_W \phi_{Oi}(W)} \quad (\text{B.13})$$

This gives us the conditional probability distribution of the window status given the observed evidence.

B.5 RCA heuristic search algorithms

The principle of the two most significant RCA types found in the literature are detailed in the following section.

B.5.1 Adtributor

Ranjita et al.(2014) proposed the Adtributor algorithm. The application scenario of the article is the anomaly analysis of advertising revenue. Adtributor assumes that the root cause can be only one attribute that is wrong, which greatly simplifies the problem, but obviously does not cover the actual needs. Adtributor used the auto-regressive and moving average model (ARMA) model for key performance indicators (KPI) prediction. What is important to note here is that KPI as indicator can be divided into two types. Classic KPI with additive quantities such as number of successes, and derived KPI which derived from

classic KPI such as success rate. Meanwhile, Adtributor proposes two metrics to evaluate the combination of attributes. It proposes explanatory power which can be defined as the percentage of change in the overall value of the measure that is explained by change in the given element's value. The equation shows below:

$$EP_{ij} = (A_{ij}(m) - F_{ij}(m)) / (A(m) - F(m)) \quad (\text{B.14})$$

In the equation, $A_{ij}(m)$ and $F_{ij}(m)$ are the actual and predicted value for an element j in dimension i separately. $A(m)$ and $F(m)$ are the actual and predicted total number of searches separately.

Then based on the Jensen-Shannon divergence, the author proposed the surprise which can be defined as a dimension that has large change in its distribution that is more likely to be a root-cause than the dimension that does not exhibit such a change. The equation is given in the following:

For the element E_{ij} , the p_{ij} is denoted as the forecasted or prior probability value

$$p_{ij}(m) = F_{ij}(m) / F(m), \forall E_{ij} \quad (\text{B.15})$$

The q_{ij} is denoted as the actual or posterior probability value

$$q_{ij}(m) = A_{ij}(m) / A(m), \forall E_{ij} \quad (\text{B.16})$$

Then the difference of two distribution is measured by Jensen-Shannon divergence D_{JS} , which is a number between 0 and 1: $0 \leq D_{JS} \leq 1$, meaning the two distributions are the same, the larger the value the greater the difference.

$$D_{JS}(P, Q) = 0.5 \left(\sum_i p_i \log \frac{2p_i}{p_i + q_i} + \sum_i q_i \log \frac{2q_i}{p_i + q_i} \right) \quad (\text{B.17})$$

Then the surprise of S_{ij} of E_{ij} is computed as following:

$$S_{ij}(m) = 0.5 * \left(p \log \frac{2p}{p + q} + q \log \frac{2q}{p + q} \right) \quad (\text{B.18})$$

Author used these two metrics to quantify the definition of the root cause. Finally, the algorithm ranks the dimensions by calculating the dimensional surprise (the sum of the surprise of all elements within the dimension) and determines the dimension in which

the root cause is located. The explanatory power of each element is calculated within the dimension, and when the sum of the explanatory powers of the elements exceeds a threshold, these elements are considered root causes.

In addition, Adtributor's root cause analysis relies heavily on the overall KPI changes. It does not perform well for data with little overall change, but it is convenient for more drastic internal fluctuations. It is not very suitable for KPIs of derived categories.

Adtributor first introduced explanatory power and surprise, which makes root causes quantifiable and has strong implications. The assumption of limiting the root cause to one dimension also greatly simplifies the complexity of the problem, but such an assumption does not quite fit our actual scenario, and it is not entirely reasonable to measure the root cause by the magnitude of the explanatory power and surprise.

Beside the Adtributor, Linnea et al.(2018) proposed the Recursive adtributor. Recursive adtributor is mainly designed to solve the problem of unreasonable assumptions of adtributor. The basic idea is to recursively use the adtributor, and in each dimension, the adtributor will return a root cause attribute and the corresponding value. The algorithm will recursively invoke the adtributor on the next selected dimension to obtain the following root cause attributes and corresponding values until finally obtaining the values of all dimensions. Lin et al. 2016 proposed the iDice which is an algorithm designed to identify the root cause of anomalies in multi-dimensional time series data, such as a sudden increase in issue reports for software systems. The algorithm employs sequentially three pruning steps: impact-based pruning, change detection-based pruning, and isolation power-based pruning. These steps reduce the search space and focus on significant attribute combinations. Finally, iDice ranks potential root causes using a score similar to the Fisher distance. While iDice effectively utilizes time series data and proposes a root cause evaluation metric, it may not significantly reduce complexity and could miss real root causes in some KPIs, particularly derived ones like success rate.

Li et al.(2019) proposed the squeeze which extends the Hotspot algorithm with a generalized ripple effect, making it applicable to both basic and derived KPIs. The algorithm employs heuristic strategies to accelerate the root cause search process while maintaining performance, clustering fine-grained attribute combinations, and identifying anomalous KPIs corresponding to the root cause.

B.5.2 Hotspot

Before introducing the Hotspot, some terms have been defined in the table B.1 for clear understanding.

Table B.1 – Definitions and Notations

Term	Definition	Notation	Example
Attributes	The categories of the information of each sensor record	-	Temperature(T), Outside
Attributes values	The candidate values of each attribute	-	Temperature(C), ... {20°C,21°C,23°C,etc.} for
Element	A combination vector of distinct values of each attribute	$e=(t,c,n,l,h)$	Temperature (20°C,*,*,*,*), (*,4kwh,*,*,*), (20°C,4kwh,*,*,*)
Value number	Heat consumption as KPI	$V(e_i)$	
Forecast value number	Forecast heat consumption as KPI	$f(e_i)$	
Data cube	A data structure of multi-dimensional data	n-d cube	A 4-d data cube with the dimensions {OU,N,S,E}
Cuboids	A cuboid is a data cube whose dimensions are in a subset of all dimensions	B_i	(B_ou B_s B_N B_E)
Potential score	A concept of measuring the potential of a set of elements to be the root cause	ps	

Sun et al.(2017) proposed the Hotspot algorithm. The application scenario in their article is web page views. The article uses the difference between the actual and predicted values of web page views to determine the anomaly and then further locates the dimension where the anomaly occurs. In the article, it uses four main dimensions: province, operator, data center, and channel. Hotspot and Adtributor use the same strategy, which is prediction plus search, to propose a root cause determination method based on ripple effect for basic additive types of KPI (page view, transaction volume, etc.). The ripple effect describes how the KPI change of the root cause node propagates to its elements. Its purpose is that when we assume that an anomaly occurs in an element, then all the children of that element should change accordingly, and the amount of change (true value - predicted value) of that element needs to be distributed to all its descendants in proportion of the predicted value. The likelihood of a factor being the root cause is determined by how much it satisfies the ripple effect condition when considered in combination with its dependent variables or "children." In other words, if the dimensional combinations with its children exhibit a strong ripple effect, it's more likely that the factor in question is the actual root cause. The ripple effect equation is shown below:

$$v(x'_i) = f(x'_i) - h(x) \times \frac{f(x'_i)}{f(x)}, (f(x) \neq 0). \quad (\text{B.19})$$

In the equation, x'_i is denoted as the descendants of x , when the page view value of x changes by $h(x)$, i.e., $h(x) = f(x) - v(x)$, $f(x)$ denoted as the predicted page view number of an element, $v(x)$ denoted as the number of access logs according to an element. x'_i will get its share of $h(x)$ according to the proportions of their forecast values.

Hotspot therefore proposes a potential score to quantify the degree to which a node satisfies the ripple effect among all its leaf nodes. The possibility that the set of elements is a root cause is determined by comparing the similarity (Euclidean distance) between the vector of derived values from ripple effect and the vector of true values. The equation is shown below:

$$ps = \max \left(1 - \frac{d(\text{real}, \text{deduced})}{d(\text{real}, \text{forecast})}, 0 \right) \quad (\text{B.20})$$

$$d(\vec{u}, \vec{w}) = \sqrt{\sum_i (u_i - w_i)^2} \quad (\text{B.21})$$

Beside, Hotspot is very different from other work such as iDice in that it considers root

cause combinations in the same dimension, e.g. not only elements such as (province1, operator1), but also elements such as (Province1, Province2) such elements, which of course brings additional search complexity and the search space will be greatly increased. So the authors propose a pruning strategy based on Monte Carlo tree search and hierarchical pruning, which searches according to the path that obtains the maximum potential score gain, and prunes the children nodes of an element if its potential score is low(hierarchical pruning).

Hotspot proposes an inspired root cause judgment method ripple effect, and innovatively applies Monte Carlo tree search to pruning, which greatly reduces the complexity of search, but Hotspot also has some limitations and does not take into account non-additive KPI, such as success rate.

Li et al.(2019) proposed the squeeze which extends the Hotspot algorithm with a generalized ripple effect, making it applicable to both basic and derived KPIs. The algorithm employs heuristic strategies to accelerate the root cause search process while maintaining performance, clustering fine-grained attribute combinations, and identifying anomalous KPIs corresponding to the root cause.

B.6 Real case BN result

The BN results for the real case study are shown in Figure B.4 and Figure B.5, Figure B.4 shows the inferred probability for window status from 0 to 4 which indicate the number of window opened. The inferred probability can be seen in the Table B.2. And Figure B.5 shows the real and predicted window status.

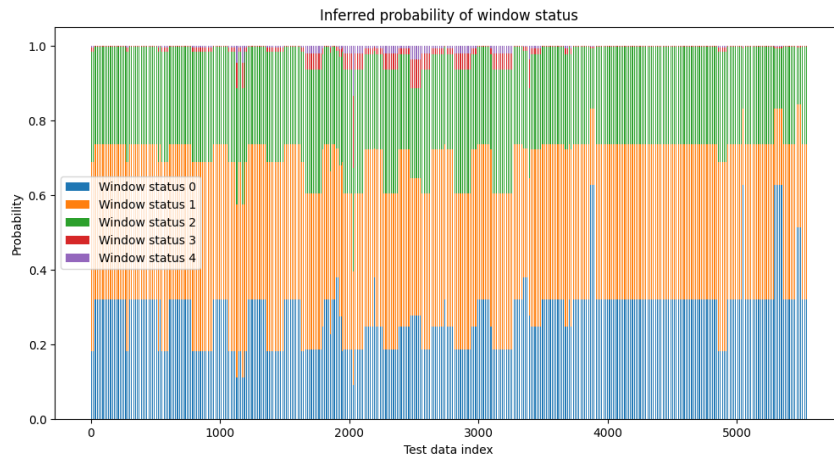


Figure B.4 – BN for real case window status inference

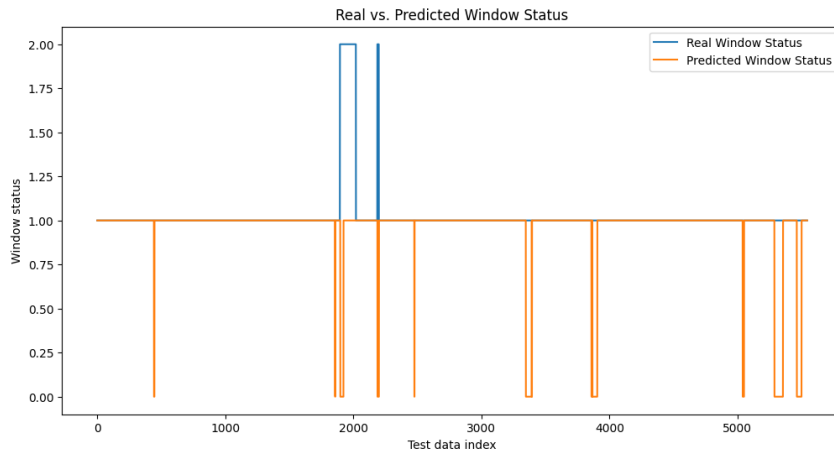


Figure B.5 – Real case window status VS inference

Table B.2 – Merged Inferred Probabilities and Ground Truth

Sample	Window Status		Occupancy Status		Ground Truth	
	Status 0	Status 1	Status 0	Status 1	Window	Occupancy
0	0.589319	0.410681	0.689737	0.310263	0	0
1	0.589319	0.410681	0.689737	0.310263	0	0
2	0.589319	0.410681	0.689737	0.310263	0	0
...
68	0.331055	0.668945	0.573473	0.426527	1	1
69	0.331055	0.668945	0.573473	0.426527	1	1
70	0.331055	0.668945	0.573473	0.426527	1	1
...

ANNEX ON PERSPECTIVES

C.1 Methodology proposal for model predictive control in this study

Model Predictive Control (MPC) is an advanced method of process control widely used in industry. It has been in use in some form or other since the early times of automatic control theory. MPC uses a model of the system to predict future outcomes and optimize the current control input based on these predictions.

The principle of MPC is to use a mathematical model of the system to predict the future behavior of the system, and then solve an optimization problem to find the best control action. The optimization problem is usually formulated as:

$$\min_u J(x, u) = \sum_{k=0}^{N-1} (x(k) - x_{\text{ref}}(k))^T Q (x(k) - x_{\text{ref}}(k)) + (u(k) - u_{\text{ref}}(k))^T R (u(k) - u_{\text{ref}}(k)) \quad (\text{C.1})$$

subject to:

$$x(k+1) = Ax(k) + Bu(k), \quad k = 0, \dots, N-1 \quad (\text{C.2})$$

where x is the state, u is the control input, x_{ref} and u_{ref} are the references for the state and control input, Q and R are the weighting matrices, A and B are the state and input matrices of the system, and N is the prediction horizon.

The optimization problem finds the sequence of control inputs u that minimize the cost function $J(x, u)$ while satisfying the system dynamics.

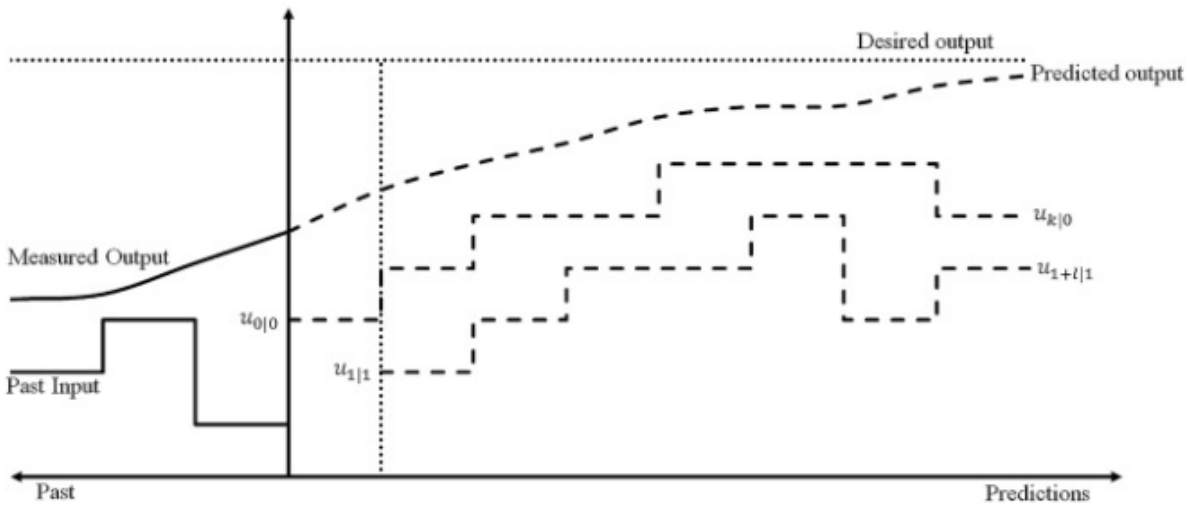


Figure C.1 – MPC example

MPC has proven its effectiveness in various fields such as process control, automotive control, and energy-efficient building control.

Model Predictive Control (MPC) is a powerful control strategy that uses a model of the system to predict future behavior and optimize the current control input. It is an effective solution for managing complex systems with constraints and interactions between variables.

C.2 Building model

Model Predictive Control (MPC) is an optimization strategy that relies heavily on first-principle models, like white-box models, which are highly accurate but mathematically complex. These models need a lot of building data, processing capacity, and upkeep because they are based on physical rules (such as equations for the balance of energy, mass, and momentum). They take a lot of time and are therefore not always appropriate for MPC in buildings. It should be noted that BPS software like Modelica, EnergyPlus, and IDA frequently uses these models. However, data-driven MPCs have begun to gain popularity as a result of the expanding usage of sensors in buildings. Black-box models or grey-box models are frequently used by these MPCs. Black-box models are purely data-driven approaches that use statistical regression or artificial neural network (ANN) techniques to analyze measurable time-series data from the system. These models require

a large amount of training data to ensure their quality, and the performance of the model can be greatly impacted by the quality of the training data.

On the other hand, grey-box models combine the fundamentals of both white-box and black-box models. They construct the model structure using the dominant physical process of the system and then use measurement data to fit the model's parameters. For applications including load estimation, optimal control, and building-grid integration, the grey box has shown its outstanding ability.

In the building context, Resistance and capacitance models (i.e., RC models) are a popular approach to creating the structure of grey-box models which effectively simulate building thermal dynamics. Then, the thermal behavior of the building is represented using an analogy to an electrical circuit, where the thermal resistance R is akin to electrical resistance, and thermal capacitance C is similar to electrical capacitance. In the model, thermal resistance stands for the obstruction to heat transfer between various building components or zones, such as walls, roofs, or internal and external surroundings. On the other hand, the thermal capacitance designates the capacity to store thermal energy of a specific building component (such as walls, air, or furniture). These elements could come together to create a network that replicates the structure's thermal behavior. Therefore, it's asserted that grey-box models demonstrate superior extrapolation capabilities when contrasted with black-box models. Yang et al.(2019) used an adaptable and reliable forecasting model to optimize the indoor environment. This model is based on a thorough grey-box model whose parameters are updated everyday. Another study by Zhang et al.(2022) created an adaptive grey-box model for buildings by introducing a technique for predicting a time-dependent solar aperture based on B-splines. Freund et al.(2019)used the R4C3 model, a grey-box model with three capacitances and four resistances to describe a thermal zone. This model was further developed with the inclusion of a thermally activated building system (TABS) model that incorporated an additional capacitance, two resistances, and an air-handling unit (AHU) resistance. By including an extra node to reflect interior thermal inertia (such as room air, furniture, etc.), Hedegaard et al.(2020) improved the ISO 13790 standard and increased the model's prediction powers under dynamic operating settings. In order to forecast thermal reactions in occupied buildings. Hedegaard et al.(2017) proposed various model structures in order to estimate the parameters of the grey-box model, effectively characterizing a building's thermal properties. The first model, a 2R2C model, took into account the thermal inertia of both the wall and air, with solar gains calculated via the effective window area and solar irradiance.

They expanded this model with the 3R2C, 4R3C, and 4R3Cw models, which progressively added heat loss resistance, interior capacity, and third thermal mass node respectively. The 3R2C and 4R3Cw models demonstrated accurate and consistent estimates across all datasets. Hu et al. (2017) applied a 5R4C grey-box model to predict room temperature, incorporating a more comprehensive understanding of room thermal dynamics. Given the Hong Kong climate, most residential buildings use lightweight wall and roof materials. As such, their model considered the external wall as one thermal resistance and two equal thermal capacitances, with an additional two capacitances accounting for indoor air and internal mass. Harb et al. (2016) developed a number of grey-box models that adhered to the equations established by the VDI 6007 standard for interior-exterior heat exchange and solar heat gains. In order to individually account for infiltration and window conduction gains, Blum et al. (2018) examined three grey-box model architectures and added an additional resistance to the 5R4C model.

In summary, The ability to capture the thermal dynamics of the building, computing cost, and the accessibility of measurement data are some of the criteria that influence the choice of model order. For single-zone models, second-order and third-order models are usually preferred due to their balance between accuracy and computational cost. Higher-order models may lead to overfitting problems if insufficient measurements are available. A second-order model is typically utilized as the basic model for each thermal zone in multi-zone models. Then, to connect these thermal zones, thermal resistances and occasionally capacitances are employed. For multi-zone models, the choice of model order also depends on the accessibility of measurement data. Given the variety of building kinds and structures, several model structures of the same order can exhibit acceptable prediction performance, demonstrating flexibility in model structure selection.

In the grey box mode, it also involve the parameter estimation, and finding the best values for the unknown parameters in a grey-box model involves a non-linear optimization process. With a defined set of resistances (R) and capacitances (C), the grey-box model can forecast the indoor air temperature profile. An objective function can then be used to assess the level of fit between the model's predicted data and actual measured data collected from in-home smart sensors during this optimization process. The goal of this optimization is to reduce the integrated root-mean-square error to its smallest possible value. Optimization techniques are required to find the optimal values of R and C which make the objective function reach the minimal value. Genetic algorithm (GA) is employed to identify the RC value. GA has been successfully used to solve the optimization problems

in the domain of HVAC. Arendt et al.(2018) used the GA combined with a gradient-based method to solve non-convex optimization problems for identifying the parameters of grey-box models. Smullen et al.(2014) develop a simulation optimization tool combined with GA to a building energy simulation engine to select the optimal values of parameters with the envelope to minimize energy use

C.2.1 Grey box model

Based on the review, a single zone 5R3C model can be used for this thesis as shown in FigureC.2

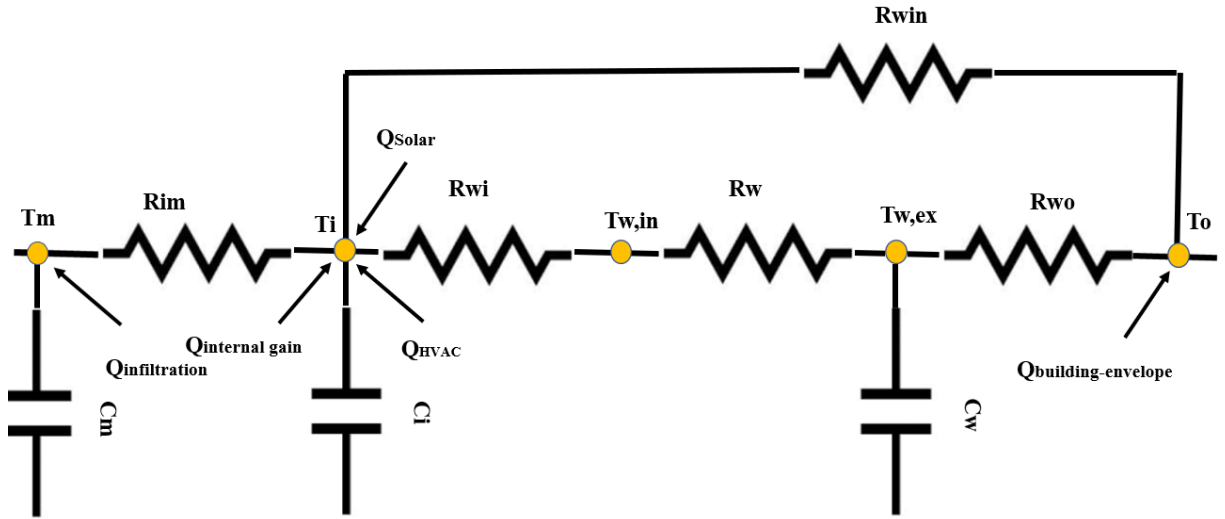


Figure C.2 – RC model 5R3C

Thermal dynamic based on Figure C.2 can be seen below,

$$\frac{dT_{w_{ex}}}{dt} = \frac{(T_o - T_{w_{ex}})}{R_{wo}} + \frac{(T_{w_{in}} - T_{w_{ex}})}{R_w} + \frac{Q_{solar_windows} + Q_{building_envelope}}{C_w} \quad (C.3)$$

$$\frac{dT_{w_{in}}}{dt} = \frac{(T_{w_{ex}} - T_{w_{in}})}{R_w} + \frac{(T_i - T_{w_{in}})}{R_{wi}} \quad (C.4)$$

$$\frac{dT_i}{dt} = \frac{(T_m - T_i)}{R_{im}} + \frac{(T_{w_{in}} - T_i)}{R_{wi}} + \frac{Q_{solar_windows} + Q_{internal_gain} + Q_{HVAC} + Q_{infiltration_ventilation}}{C_i} \quad (C.5)$$

$$\frac{dT_m}{dt} = \frac{(T_i - T_m)}{R_{im}} + \frac{Q_{internal_gain}}{C_m} \quad (C.6)$$

Within the equations and RC model. notations are given as below,

- $T_{w_{ex}}$: Exterior wall surface temperature
- $T_{w_{in}}$: Interior wall surface temperature
- T_i : Indoor temperature
- T_m : Internal mass temperature
- T_o : Outdoor temperature
- R_{wo} : Resistance between outdoor and exterior wall
- R_w : Resistance between exterior and interior wall
- R_{wi} : Resistance between interior wall and indoor
- R_{im} : Resistance between indoor and internal mass
- C_w : Capacitance of wall
- C_i : Capacitance of indoor air
- C_m : Capacitance of internal mass
- $Q_{solar_windows}$: Heat gain from solar through windows
- $Q_{building_envelope}$: Heat transfer through building envelope
- $Q_{internal_gain}$: Internal heat gain from sources like computers, occupants, etc.
- Q_{HVAC} : Heat provided by HVAC system
- $Q_{infiltration_ventilation}$: Heat loss or gain due to infiltration and ventilation

C.2.2 Parameter estimation

To improve the accuracy and computation efficiency of the optimizations, the optimal parameters can be accurately and efficiently obtained by following procedure

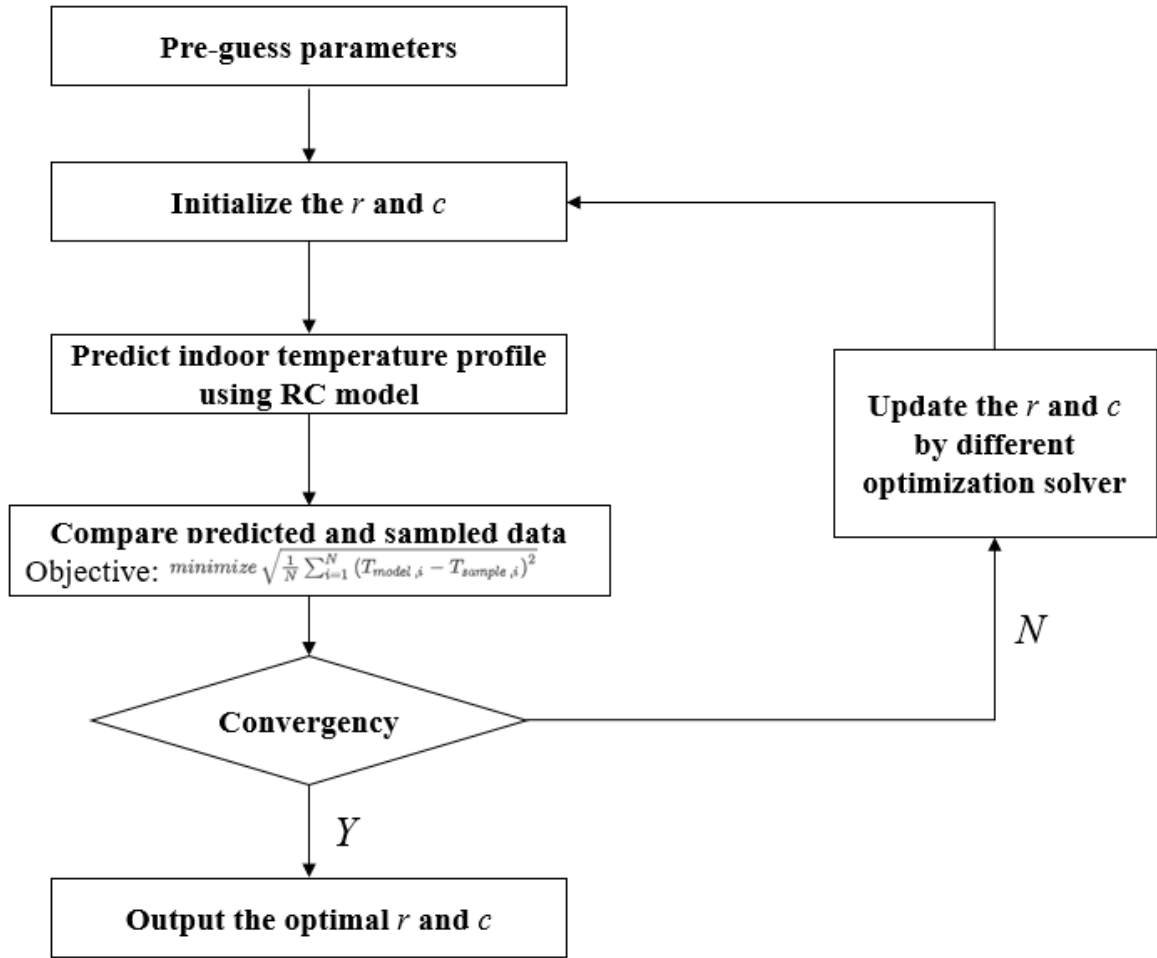


Figure C.3 – Parameter estimation flow chart

$$J(C_w, C_i, C_m, R_{win}, R_w, R_{w,o}, R_{w,i}, R_{i,m}) = \text{minimize } \sqrt{\frac{1}{N} \sum_{i=1}^N (T_{model,i} - T_{sample,i})^2} \quad (\text{C.7})$$

Finding the best values for the unknown parameters in a grey-box model involves a non-linear optimization process. With a defined set of resistances (R) and capacitances (C), the grey-box model can forecast the indoor air temperature profile. An objective function can then be used to assess the level of fit between the model's predicted data and actual measured data collected from in-home smart sensors during this optimization process. The goal of this optimization is to reduce the integrated root-mean-square error to its

smallest possible value. Optimization techniques are required to find the optimal values of R and C which make the objective function reach the minimal value. Genetic algorithm (GA) is employed to identify the RC value. GA has been successfully used to solve the optimization problems in the domain of HVAC. Arendt et al., used the GA combined with a gradient-based method to solve non-convex optimization problems for identifying the parameters of grey-box models. Daniel et al.2010 develop a simulation optimization tool combined with GA to a building energy simulation engine to select the optimal values of parameters with the envelope to minimize energy use.

C.2.3 Model validation

After applying the Genetic Algorithm (GA) for model optimization, validating the model's performance on unseen data becomes crucial.

1. Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |T_{\text{model},i} - T_{\text{sample},i}|$$

MAE calculates the average absolute difference between the predicted and actual values. It provides a general sense of the model's accuracy.

2. Mean Absolute Percentage Error (MAPE)

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \left| \frac{T_{\text{model},i} - T_{\text{sample},i}}{T_{\text{sample},i}} \right|$$

MAPE calculates the average of the absolute percentage errors. It's useful when you want to understand the error in terms of the relative size compared to the actual values.

3. Root Mean Squared Error (RMSE)

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (T_{\text{model},i} - T_{\text{sample},i})^2}$$

RMSE is the square root of the average of the squared differences between the predicted values and the actual values. By squaring the differences and then taking the square root, RMSE gives a higher weight to larger errors..

C.3 User information integration

As mentioned in the beginning of this Chapter, MPC has its advantage which can considering different constraints. With the detected user information from our research, we can transfer our information to a user schedules, this schedule can be predefined or real time updated. The user schedules could include information such as occupied or unoccupied. And within each building state, we can given state constrains that are in terms of lower and upper temperature bounds. At any time step k , these constraints are given by:

$$X_k = [T_{\text{lower}}, T_{\text{upper}}] \quad (\text{C.8})$$

where:

- T_{lower} is the lower temperature bound.
- T_{upper} is the upper temperature bound.

During occupied states, the constraints are:

$$X_k = X_{\text{comfort}} = [T_{\text{comfort_lower}}, T_{\text{comfort_upper}}] \quad (\text{C.9})$$

There's a potential to save energy by relaxing these constraints during unoccupied times, allowing for broader temperature ranges. During unoccupied states, the constraints are:

$$X_k = X_{\text{setback}} = [T_{\text{setback_lower}}, T_{\text{setback_upper}}] \quad (\text{C.10})$$

Temperature constraints should be set dynamically based on predicted occupancy. If p_k is the occupancy probability at time k from BN:

$$\begin{aligned} &\text{if } p_k < th_l \text{ then } X_k = X_{\text{setback}} \\ &\text{if } p_k > th_u \text{ then } X_k = X_{\text{comfort}} \end{aligned}$$

Where th_l and th_u are the lower and upper thresholds for occupancy prediction respectively. For probabilities between th_l and th_u , a smooth function such as a sigmoid can be used:

$$X_k = f(p_k; \text{parameters}) \quad (\text{C.11})$$

where f represents the function used. Then, the constraints can be integrated into MPC, the objective of the MPC is to minimize the cost over a prediction horizon N :

$$J = \sum_{i=1}^N (x_i^T Q x_i + u_i^T R u_i) \quad (\text{C.12})$$

Subject to:

- System dynamics: $x_{i+1} = Ax_i + Bu_i$
- Control constraints: $u_{\min} \leq u_i \leq u_{\max}$
- State constraints: $x_{\min} \leq x_i \leq x_{\max}$

For a building system, the state x_i represents the temperature, and u_i is the control input (e.g., heating or cooling amount). The temperature constraints can be integrated as:

$$T_{\text{lower}}(i) \leq x_i \leq T_{\text{upper}}(i) \quad (\text{C.13})$$

Where:

- $T_{\text{lower}}(i)$ and $T_{\text{upper}}(i)$ are the temperature bounds at time i .
- These bounds can be either the comfort or setback bounds, based on predicted occupancy or a predefined schedule.

If the predicted occupancy probability at time i is above a certain threshold th_u :

$$\begin{aligned} T_{\text{lower}}(i) &= T_{\text{comfort_lower}} \\ T_{\text{upper}}(i) &= T_{\text{comfort_upper}} \end{aligned}$$

Otherwise:

$$\begin{aligned} T_{\text{lower}}(i) &= T_{\text{setback_lower}} \\ T_{\text{upper}}(i) &= T_{\text{setback_upper}} \end{aligned}$$

Above shows example illustration for integrating the user information into MPC, but in real application, it should be much more complexity, since there are interaction of different activities. And many more constraints need to be considered into the MPC in order to guarantee the energy save and comfort.

Titre : Vers des approches d'optimisation centrée sur l'utilisateur et de contrôle prédictif de la performance des bâtiments intelligents

Mot clés : Maison intelligente, Apprentissage automatique, Optimisation du placement de capteur, Détection des actions des occupants

Résumé : En réponse aux transitions énergétique et numérique dans le secteur du bâtiment, le concept de Smart Building gagne en importance. La gestion optimisée de ces bâtiments devrait permettre une réduction des coûts et des consommations énergétiques en phase d'exploitation. Cependant, l'amélioration des performances énergétiques ne doit pas se faire au détriment du confort des usagers. Le succès d'un système de gestion intelligente des bâtiments dépend en grande partie de la participation active de ses occupants et il est important de prendre en compte les "humains dans la boucle". De plus, pour des raisons économiques et environnementales, le déploiement de capteurs et actionneurs intelligents doit se faire avec parcimonie. Dans ce contexte, l'objectif de la thèse est d'optimiser le placement de capteurs multi-physiques pour détecter une ou plusieurs actions de l'occupant qui serviront à développer des stratégies de gestion efficaces centrées sur l'utilisateur.

Une détection correcte de l'occupation peut nécessiter l'utilisation de nombreux capteurs intelligents mesurant diverses grandeurs physiques. De plus, les quelques ensembles de capteurs sélectionnés doivent être adaptés pour décrire le confort à certains endroits précis du bâtiment où les occupants sont plus susceptibles de se trouver. Le placement optimal des capteurs a été étudié en utilisant la méthode indépendante efficace afin d'assurer une caractérisation parcimonieuse et efficace de l'occupation et du confort (chapitre 3).

L'inclusion des utilisateurs nécessite une meilleure compréhension de leurs activités.

Des algorithmes d'apprentissage automatique sont ensuite appliqués pour détecter soit les ouvertures de fenêtres, soit la présence d'occupants dans une pièce. L'un des défis concerne l'efficacité de la détection d'activité pour les données non étiquetées (c'est-à-dire lorsque le statut d'occupation réel est inconnu). Elle a été abordée à l'aide d'algorithmes non supervisés ou semi-supervisés (chapitre 4). Un autre défi consiste à déterminer quel type d'activité est associé à différents effets sur les conditions environnementales intérieures. Pour traiter cet aspect, il a été proposé dans la thèse d'utiliser des méthodes d'analyse des causes racines et plus particulièrement les réseaux bayésiens (chapitre 5).

Une fois que l'occupation est mieux comprise à partir d'un ensemble réduit de capteurs, une stratégie de gestion plus efficace et centrée sur l'utilisateur peut être définie.

La cohérence du cadre proposé a été prouvée en appliquant les méthodes sur deux études de cas complémentaires. Le premier est une étude de cas simulée sur laquelle des simulations énergétiques dynamiques de bâtiments et des simulations CFD sont effectuées pour évaluer les températures intérieures et les charges énergétiques pour différents scénarios d'occupation prédéfinis. Pour la deuxième étude de cas, une instrumentation a été conçue et déployée dans un bâtiment réel : les capteurs multi-physiques intelligents installés permettent de collecter des données sur l'environnement intérieur, ainsi que sur l'occupation.

Title: Towards user-centric optimisation and predictive control approaches of the performance of smart buildings

Keywords: Smart house, Machine learning, Optimal sensor placement, Occupants activity detection

Abstract: In response to the energy and digital transitions in the building sector, the concept of Smart Building is gaining importance. The optimized management of these buildings should enable a reduction in costs and energy consumption in the operating phase. However, the improvement of the energy performances should not be at the expense of user comfort. The success of a smart house system largely depends on the active participation of its occupants and it is important considering the "humans in the loop". In addition, for economic and environmental reasons, the deployment of smart sensors and actuators must be done sparingly. In this context, the aim of the PhD is to optimize the placement of multi-physical sensors to detect one or more occupant's actions that will serve to develop efficient user-centric management strategies.

A correct detection of occupancy may require using a lot of smart sensors measuring various physical quantities. Furthermore, the few sets of sensors selected must be adapted to describe the comfort at some specific places in the building where the occupants are more likely to be. Optimal sensor placement has been investigated using the effective independent method in order ensure a sparse and efficient characterization of the occupancy and comfort (chapter 3).

The inclusion of humans in the loop requires a better understanding of their activi-

ties. Machine learning algorithms are then applied to detect either windows openings or the presence of occupants in a room. One of the challenges concerns the efficiency of activity detection for unlabeled data (i.e., when the actual occupancy status is unknown). It has been approached using unsupervised or semi-supervised algorithms (chapter 4). Another challenge is to classify which type of activity is associated with different effects on indoor environmental conditions. To deal with this aspect, it has been proposed in the thesis to use self learning algorithm and root cause analysis methods and more particularly Bayesian networks (chapter 5).

Once the occupancy is better understood from a reduced set of sensors, this knowledge is intended to be used to define an efficient user-centric energy management strategy.

The consistency of the proposed framework was proven by applying the methods on two complementary case studies. The first one is a simulated case study on which dynamic building energy simulations and computational fluid dynamics simulations are performed to assess indoor temperatures and energy loads for different predefined occupancy scenarios. For the second case study, an instrumentation has been designed and deployed in a real building: the installed smart multi-physical sensors allow collecting data on the indoor environment, as well as on the occupancy.