KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Technology

Contactless acoustic based indoor monitoring systems

A study on different acoustic models and learning strategies

Lode Vuegen

Supervisors:
Prof. dr. ir. B. Vanrumste
Prof. dr. ing. P. Karsmakers
Prof. dr. ir. H. Van hamme

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Technology (PhD)

October 2019

Contactless acoustic based indoor monitoring systems

A study on different acoustic models and learning strategies

Lode VUEGEN

Examination committee:
Prof. dr. ir. L. Appels, chair
Prof. dr. ir. B. Vanrumste, supervisor
Prof. dr. ing. P. Karsmakers, supervisor
Prof. dr. ir. H. Van hamme, supervisor
Prof. dr. ir. H. Blockeel
Prof. dr. J. Vennekens
Dr. ir. ing. T. Croonenborghs
(Wayfair, United States)
Prof. dr. ir. R. Kleihorst
(SENSO2ME, Belgium)
Dr. MSc. R. Serizel
(Loria, France)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Technology (PhD)

October 2019

© 2019 KU Leuven – Faculty of Engineering Technology
Uitgegeven in eigen beheer, Lode Vuegen, Kleinhoefstraat 4, B-2440 Geel (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm, electronic or any other means without written permission from the publisher.

Voorwoord

Beste lezer,

Deze thesis is geschreven voor het behalen van de graad Doctor in de industriële ingenieurswetenschappen aan de KU Leuven. Gedurende een periode van 7 jaar heb ik hier met hart en ziel aan gewerkt om dit succesvol af te ronden. Nu wil ik even de tijd nemen om enkele belangrijke personen te bedanken die mij hierin gesteund en geholpen hebben.

Eerst en vooral wil ik mijn promotoren prof. dr. ir. Bart Vanrumste, prof. dr. ir. Hugo Van hamme en prof. dr. ing. Peter Karsmakers bedanken. Ze hebben mij vanaf dag één onder de vleugels genomen om mij zo goed mogelijk te begeleiden. Zonder hun begeleiding en deskundig advies was het voor mij een onmogelijke opdracht geweest om dit doctoraat tot een goed einde te brengen.

Naast het technische advies van mijn promotoren stonden mijn ouders dag op dag weer klaar voor de mentale ondersteuning. Gedurende mijn periode als doctorandus hebben ze mij meermaals moeten aanmoedigen om vol te houden. Zonder hun schouderklopjes zou deze thesis er uiteindelijk nooit geweest zijn.

Daarnaast wil ik ook al mijn collega's bedanken voor de aangename werksfeer met de nodig humor. Dit heeft ertoe geleid dat ik dag op dag steeds weer gemotiveerd naar het werk kwam.

Tot slot wil ik ook nog een dankwoordje richten naar al mijn vrienden. Ook zij hebben mij gesteund en aangemoedigd gedurende mijn periode als doctorandus.

Dank je wel allemaal!

Lode Vuegen

Abstract

The number of physical devices connected to the internet has increased significantly in recent years. This resulted in the concept of '*Internet of Things*' (IoT) where everyday devices (e.g. home appliances, vehicles, industrial machinery, etc.) become an entity on the internet with the ability to exchange their sensed observations. The combination of a local computing unit to process the collected information together with the ability to communicate over the network results in a '*smart behaviour*' and thereby improving the interaction with the end-user.

The use of IoT-enabled devices for indoor monitoring applications has already been widely examined in the literature during the last decade. Especially the use of relatively simple IoT sensors such as those that measure temperature, humidity and passive infrared (PIR) gained a lot of research interest. This doctoral research will focus on the use of acoustics as a sensing modality which until now received less attention for its purpose in indoor monitoring applications. Nowadays, acoustic data can be collected from the environment without the need of expensive and sophisticated hardware by simply using the built-in microphones that are available in different types of devices that we already use in our daily lives (e.g. television, home assistants, speakers, etc.). The use of acoustics for indoor monitoring applications has some interesting advantages compared to other sensing modalities. First of all, acoustic sensors can collect information from the environment at a distance and thus does not require any physical contact with the objects that are being monitored. This implies that a single acoustic sensor can be used to monitor multiple objects at once or even an entire room and thereby reducing the required number of sensors significantly. Secondly, acoustic data is highly informative and contains a lot of information about the environment being monitored. For instance, the spectral part of the data gives insight in the type of sound events that have been occurred, e.g. footsteps, speech, opening/closing doors, etc., while the spatial part can be used to extract information about the location of the sound sources. Thirdly, acoustic monitoring is unaffected by change in illumination and does not require a line-of-sight path to the sound sources being monitored. This implies that acoustic observations systems are typically less sensitive to occlusions (obstruction in the line-of-sight path) compared to video cameras or

passive infrared motion detectors.

This doctoral research has three main contributions regarding acoustic monitoring of indoor environments. This first contribution is situated in automatically classifying of activities of daily living from the available acoustic information in order to support the elderly health care. More specifically, by automatically compiling a summary report about the performed activities over a longer period of time (e.g. few weeks) allow caregivers to make an objective assessment of the health status of the elderly with minimal effort. For instance, increasing activity durations over time or frequently skipping a meal might indicate an underlying health issue. The main focus during this research situated in the use of existing supervised machine learning algorithms which have been evaluated on noise robustness and computational complexity to assess the edge integration in an IoT module. The second contribution of this doctoral research is situated in modelling and classifying of acoustic events using a '*weakly supervised*' and '*multi-label*' convolutive learning algorithm. The multi-labelling aspect makes that multiple labels (e.g. footsteps, speech, closing door, etc.) can be assigned to a larger acoustic segment (e.g. 30 seconds of sound data). The weakly supervised part implies that not all occurring events in a given segment must be labelled and that no begin nor end timestamps of the individual events need to be provided. This strategy in labelling the data reduces the workload in labelling the data significantly. The proposed algorithm is evaluated on the dependence of unlabelled events per segment, the influence of overlapping events in the training set and the robustness to background noise. The last contribution of this doctoral research is devoted the development of a '*adaptive*' acoustic presence detector. The objective of the classifier is situated in discriminating presence from absence related sound data and not in classifying individual sound events from data labelled by user specific inputs (e.g. turning on the lights, opening door, etc.). Furthermore, the computational complexity of the proposed learning strategy is also taken into account such that the presence detection algorithm can be mapped to an embedded platform.

The aforementioned contributions will be investigated in this dissertation on self-collected or publicly available real-life acoustic datasets. All examined machine learning algorithms are implemented in a technical software development environment (MATLAB) and are subsequently validated regarding their intended application. The concluding remarks will finalise this doctoral dissertation by briefly recapitulating the research trajectory and the obtained results together with a short discussion about the future research prospects and how these aspects can be tackled.

Beknopte samenvatting

Tijdens de afgelopen jaren is het aantal toestellen dat verbonden is met het internet sterk toegenomen. Deze toename van geconnecteerde toestellen heeft geleid tot de term '*het internet der dingen*' beter gekend als IoT. Alledaagse voorwerpen (bv. huishoudtoestellen, voertuigen, industriële machines, enz.) worden hierdoor een entiteit op het internet met de mogelijkheid om hun waarnemingen te communiceren met andere objecten en personen. De combinatie van lokale rekenkracht samen met de mogelijkheid om waarnemingen van andere sensoren op te vragen maakt dat er autonome beslissingen genomen kunnen worden. Deze autonomie maakt dat de toestellen een bepaald '*intelligent gedrag*' gaan vertonen waardoor een verhoogde interactie met de gebruiker bekomen kan worden.

Het gebruik van IoT sensoren voor het monitoren van huiselijke omgevingen heeft de afgelopen jaren reeds veel aandacht gekregen. Hiervoor heeft men vooral gefocust op het gebruik van relatief eenvoudige sensoren zoals temperatuur, luchtvochtigheid, illuminatie, passief infrarood (PIR), enzovoort. In dit proefschrift zal de focus gelegd worden op het gebruik van de tot nu toe weinig belichte audiomodaliteit. De dag van vandaag kan er op eenvoudige wijze akoestische informatie van de omgeving verzameld worden zonder de nood van dure en gesofisticeerde acquisitiesystemen. Denk hiervoor bijvoorbeeld aan akoestische sensoren dewelke tegenwoordig standaard geïntegreerd zijn in televisies, radio's, computers, enzovoort. Het gebruik van akoestiek heeft enkele interessante voordelen t.o.v. andere modaliteiten. Ten eerste, akoestische sensoren zijn contactloos en kunnen dus op afstand de omgeving monitoren. Dit maakt dat één akoestische sensor meerdere toestellen kan monitoren waardoor het aantal benodigde sensoren per ruimte beperkt blijft. Ten tweede, akoestische signalen zijn zeer rijk aan informatie. Zo is er enerzijds de spectrale informatie dewelke inzicht geeft over het soort geluid dat zich heeft voorgedaan, bv. voetstappen, spraak, alarm, enz., en anderzijds is er de spatiale informatie dewelke inzicht geeft over de locatie van de geluidsbron. Ten derde, in tegenstelling tot (infrarood) camera's zijn akoestische sensoren niet gevoelig aan externe belichting en vereisen ze geen gezichtslijn tussen de geluidsbron en de sensor. Dit maakt dat akoestische sensoren ook minder gevoelig zijn aan obstructies.

Dit doctoraatsonderzoek heeft drie voorname contributies m.b.t. het akoestisch monitoren van binnenhuis omgevingen. De eerste bijdrage is gesitueerd in het automatisch herkennen van activiteiten van het dagelijkse leven gebruikmakende van de aanwezige akoestische informatie. Dit met als oogpunt om de ouderenzorg te ondersteunen. Meer specifiek, door het genereren van objectieve rapporten over de uitgevoerde activiteiten over een langere periode kan de zorgverlener een inschatting maken over de gezondheidstoestand van de oudere in kwestie. Tijdens dit onderzoek werd de focus gelegd op het gebruik van bestaande gesuperviseerde machinale leeralgoritmes dewelke geëvalueerd zijn op ruisrobuustheid en computationele complexiteit om de integratie naar een IoT module te onderzoeken. De tweede bijdrage van dit onderzoek situeert zich in het modelleren en classificeren van akoestische gebeurtenissen gebruikmakende van een '*zwak-gesuperviseerd*' en '*multi-label*' convolutief leeralgoritme. Het kernwoord multi-label duidt aan dat een akoestisch segment (bv. 30 seconden aan geluid) meerdere labels meekrijgt (bv. deur dichtdoen, menselijke spraak, voetstap). Onder zwak-gesuperviseerd wordt verstaan dat de labels niet volledig zijn (niet alle gebeurtenissen die voorkomen in het segment hoeven gelabeld te zijn) en dat er ook geen aanduiding is wanneer de akoestische gebeurtenis zich exact voordoet in het gegeven segment. Dit soort labels heeft als voordeel dat het annotatieproces (dat vaak uitgevoerd wordt door mensen) veel vlotter kan verlopen. Het voorgestelde convolutief leeralgoritme wordt geëvalueerd op de afhankelijkheid van het aantal akoestische gebeurtenissen zonder label per segment alsook naar de ruisrobuustheid en de invloed van overlappende gebeurtenissen in de training set. De laatste bijdrage van dit doctoraatsonderzoek is gesitueerd in het automatisch herkennen van persoonsaanwezigheid in een ruimte op basis van bijhorende akoestische informatie. Hiervoor hebben we gebruik gemaakt van een '*adaptieve*' leerstrategie die automatisch de interessante geluiden gaat leren dewelke gelinkt kunnen worden aan aanwezigheid of afwezigheid aan de hand van data gelabelled door inputs van de gebruiker zelf (bv. manueel inschakelen van de verlichting, openen/sluiten van een deur, enz.). Tevens is er ook rekening gehouden met de computationele complexiteit van het algoritme zodanig dat het geïmplementeerd kan worden op een ingebed platform.

Tijdens dit proefschrift zijn de bovenvermelde bijdragen toegepast op zelf opgenomen of reeds publiek beschikbare levensechte audiodata. Alle gebruikte machinale leeralgoritmes zijn geïmplementeerd in een technische softwareomgeving (MATLAB) en vervolgens gevalideerd naar hun beoogde toepassing. Ter afronding van dit proefschrift wordt er ook nog een algemene conclusie gegeven over het uitgevoerde onderzoek en de daarbij horende resultaten. Tot slot wordt er ook nog kort de aandacht geschonken aan de verdere onderzoekslijnen en hoe deze aspecten mogelijks gerealiseerd kunnen worden.

Contents

Abstract iii

Contents vii

List of Abbreviations xiii

List of Symbols and Notations xvii

List of Figures xix

List of Tables xxv

1 Introduction 1

1.1 The era of Internet of Things and Big Data 2

1.2 Relevance of acoustic indoor monitoring 5

1.3 Current trends and their bottlenecks 7

1.4 Aim of this research 12

1.5 Chapter by chapter overview 15

2 Main aspects of supervised acoustic classification 19

2.1 Introduction 20

2.2 From raw data to acoustic features 21

2.2.1 MEL-frequency Cepstral Coefficients 22
2.3 Baseline acoustic classifiers 26
2.3.1 Gaussian mixture model 26
2.3.2 Support vector machine 29

3 Supervised models for classifying activities of daily living 33
3.1 Introduction 35
3.2 Collected dataset 36
3.2.1 Recording system 36
3.2.2 Home environment 37
3.2.3 Recorded dataset 38
3.3 System architecture 38
3.3.1 Sound activity detection and signal-to-noise ratio estimation 39
3.4 Acoustic feature extraction 41
3.4.1 Acoustic classification 43
3.5 Background noise simulation environment 44
3.6 Experiments and results 45
3.6.1 Experimental setup 45
3.6.2 Clean conditions 46
3.6.3 Noisy conditions 49
3.7 Conclusions 52

4 Weakly supervised multi-label convolutive event modelling and classification 55
4.1 Introduction 57
4.2 Convolutive event modelling 59
4.2.1 Introduction to non-negative matrix deconvolution . . . 59
4.2.2 Low-resolution multi-label NMD 61

4.2.3 Finding activations from the basis data 65
4.3 Experimental specific details 66
4.3.1 From raw data to features 66
4.3.2 Basis initialisation 67
4.3.3 Background modelling 68
4.3.4 Event classification 68
4.4 Experiments and results 69
4.4.1 Experimental setup 69
4.4.2 Baselines 70
4.4.3 Single-label learning mode 71
4.4.4 Low-resolution multi-label learning mode 77
4.5 Discussion 83
4.6 Conclusion 85

5 On-edge adaptive acoustic presence detection 87
5.1 Introduction 88
5.2 System architecture 90
5.3 On-edge presence detection classifiers 96
5.3.1 Support vector machine 96
5.3.2 Least-squares SVM 97
5.3.3 Fixed-size least-squares SVM 98
5.3.4 Subset-of-regressors least-squares SVM 100
5.4 Adaptive learning 101
5.4.1 Normal step 102
5.4.2 Growing step 103
5.4.3 Pruning step 104
5.5 Experimental specific details 105

5.5.1 Recorded dataset 105
5.5.2 Training and test set generation 107
5.5.3 Evaluation metric 107
5.5.4 Learning strategies 108
5.6 Results 110
5.6.1 Offline learning 110
5.6.2 Adaptive learning using a fixed-model architecture . . . 112
5.6.3 Adaptive learning using an adaptive-model architecture 114
5.7 Discussion 117
5.8 Conclusion 118

6 Conclusion and future work **121**
6.1 Summary of results 122
6.2 Future research 126
6.2.1 Research towards scientific challenges 126
6.2.2 Research towards practical challenges 128

7 Appendices **131**
7.1 LRM-NMD multiplicative updates 133
7.1.1 Activations ($\mathbf{X}$) 133
7.1.2 Dictionary elements ($\mathbf{A}^{[o]}$) 134
7.1.3 Labelling matrix ($\mathbf{A}^{[s]}$) 135
7.2 LRM-NMD results 136
7.2.1 Single-label learning mode results 136
7.2.2 Low-resolution multi-label results 137
7.3 Relation kernel matrix and feature map 139
7.4 FS-LSSVM normal step updates 140

Bibliography **143**

List of publications **161**

Biography **165**

List of Abbreviations

Abbreviation	Description
1v1	One-versus-one
1vA	One-versus-all
A/D	Analog-to-digital
ADL	Activity of daily living
AED/C	Acoustic event detection/classification
AGC	Automatic gain control
AL	Active learning
ANN	Artificial neural network
ARD	Automatic relevance determination
ARM	Acorn RISC machine
CNMF	Convolutive non-negative matrix factorisation
CNN	Convolutional neural network
CO_2	Carbon dioxide
CPU	Central processing unit
CSC	Convolutive sparse coding
DCT	Discrete cosine transform
DFT	Discrete Fourier transform
DNN	Deep neural network
DSP	Digital signal processor
EM	Expectation-maximisation
EMI	Electromagnetic interference
FFT	Fast Fourier transform
FOV	Field of view
FS-LSSVM	Fixed-size LS-SVM
GWP	Gross world product
GMM	Gaussian mixture model
IBM	International Business Machines Corporation
IoE	Internet of everything
IoT	Internet of things
L-HVAC	Lightning, heating, ventilation and air conditioning
LS-SVM	Least-squares support vector machine

Abbreviation	Description
LOS	Line-of-sight
LRM	Low-resolution multi-label
LSTM	Long short-term memory
MAP	Maximum a posteriori
MCU	Micro controller unit
MEMS	Microelectromechanical system
MFCC	Mel-frequency cepstral coefficients
ML	Maximum likelihood
NLOS	Non-line-of-sight
NMD	Non-negative matrix deconvolution
NMF	Non-negative matrix factorisation
PCA	Principal component analysis
PIR	Passive infra-red
PV	Prototype vector
ReLu	Rectified linear unit
RBF	Radial basis function
RF	Radio-frequency
RFID	Radio-frequency identification
RIR	Room impulse response
RLS	Recursive least-squares
RNN	Recurrent neural network
SAD	Sound activity detector
SNR	Signal-to-noise ratio
SSL	Semi-supervised learning
SR-LSSVM	Subset-of-regressors LS-SVM
SVM	Support vector machine
TDL	Task-driven learning
TNMF	Task-driven non-negative matrix factorisation
TVOC	Total volatile organic compounds
WASN	Wireless acoustic sensor network

List of Symbols and Notations

Matrix/vector notations	Description
$\mathbf{A}$	Matrix notation (uppercase bold)
$\mathbf{a}$	Vector notation (lowercase bold)
A	Scalar (uppercase unbold)
$\mathbf{A}_{ij}$	ij^{th} element in $\mathbf{A}$ [a]
$\mathbf{a}_i$	i^{th} element in $\mathbf{a}$ [a]

Operators	Description
$\top$	Transpose
$\lvert \cdot \rvert$	Absolute value
$\lVert \cdot \rVert_1$	L_1-norm
$\lVert \cdot \rVert_2$ and $\lVert \cdot \rVert$	L_2-norm
$\lVert \cdot \rVert_F$	Frobenius-norm
$\overbrace{(\cdot)}^{t}$	Left shift of t entries
$\overbrace{(\cdot)}^{t}$	Right shift of t entries
$\mathbf{A} \circ \mathbf{B}$	Elementwise multiplication
$\frac{\mathbf{A}}{\mathbf{B}}$	Elementwise division

[a] Unless stated otherwise.

List of Figures

1.1 The estimated number of installed IoT devices vs. the average IoT sensor cost forecast in 2020 [153, 154]. 2

1.2 Conceptual visualisation of a smart city [106]. 3

1.3 The difference between an ANN and DNN is situated in the number of hidden layers between input and output as shown in (a) and (b). The internal structure of an artificial neuron is shown in (c). 8

1.4 The conceptual difference between a traditional machine learning task and deep learning. Traditional machine learning requires a manual selection of the feature extraction and classification algorithm being used. Deep learning merges both stages in an automatic fashion without the need of explicitly defining them. 10

2.1 Block scheme of a standard supervised machine learning task. The upper part visualises the training aspect of the classifier while the lower part corresponds to the prediction phase. 21

2.2 MEL-frequency cepstral coefficients feature extraction process. The raw acoustic data is transformed to the feature domain by applying a (1) framing and windowing, (2) discrete Fourier transform, (3) MEL-frequency warping and (4) cepstral analysis on the log-transformed MEL-magnitude features. 23

2.3 The relation between the perceptual MEL-frequency scale (f_{MEL}) and the traditional frequency scale (f). The MEL-filter bank spanning 10 bands are uniformly distributed at the MEL-warped spectrum, or non-uniformly at the original frequency spectrum. 25

2.4 Graphical visualisation of a 2D input space modelled by a Gaussian mixture model containing two mixture components. . 28

2.5 Illustration of the margin of a support vector machine in case of linearly separable data. The maximum margin is defined only by the support vectors which are marked by a circle. 29

3.1 Floor plan of the home environment together with the node numbers (denoted by 1 to 7) and the two noise source locations (marked by A and B). 37

3.2 Overview of the proposed system architecture for ADL classification. Each node in the WASN determines the SNR of the received data and only the node with the highest SNR will extract the ADL relevant acoustic features. These 'best' features are then sent to the CPU to learn the model specific parameters of the classifier during training or for classification in runtime mode. . 39

3.3 Noisy SVM classification results with noise source at position A for the feature parameter setting of 20 MEL-filters with 14 cepstral coefficients (bottom) and 10 MEL-filters and 7 cepstral coefficients (top) respectively. The standard deviation of the best SNR node is also visualised in the plots. 51

3.4 Noisy SVM classification results with noise source at position B for the feature parameter setting of 20 MEL-filters with 14 cepstral coefficients (bottom) and 10 MEL-filters and 7 cepstral coefficients (top) respectively. The standard deviation of the best SNR node is also visualised in the plots. 52

4.1 Graphical visualisation of the low-resolution multi-labelling strategy (LRM). Each segment $\mathbf{Y}_j^{[o]}$ is labelled by its low-resolution multi-label vector $\mathbf{y}_j^{[s]}$ indicating the sound classes, or a subset of sound classes, that take place in the corresponding segment. As one can see, not all events in the segments must be labelled as occurring. 61

4.2 A graphical example of LRM-NMD where events of three different sound classes ($C = 3$) in $\mathbf{Y}^{[o]}$ are decomposed into a set of temporal basis matrices ($L = 3$) each four frames long ($T = 4$) and their activations. The operators $*$ and $\times$ denote the convolutive factorisation according to Eq. 4.1 and a standard matrix multiplication respectively. The upper part visualises the convolutive modelling of events w.r.t. the low-resolution multi-label vectors given in the lower part. The learned temporal basis data $\mathbf{A}^{[o]}$ can be interpreted as a set of dictionary elements describing the underlying structure in the observation data $\mathbf{Y}^{[o]}$ while $\mathbf{A}^{[s]}$ acts as a labelling matrix of $\mathbf{A}^{[o]}$ indicating to which sound class each learned dictionary element corresponds. 62

4.3 Graphical visualisation of the difference between single-label learning mode (left) vs. low-resolution multi-label learning mode (right). 69

4.4 Influence of the semantic balancing parameter (η) on the classification performance of LRM-NMD in single-label learning mode for different number of training events per sound class (n_{tr}) when $\lambda = 5$. 72

4.5 A graphical visualisation of the basis data for the classes *'drawer'*, *'toaster'* and *'locking/unlocking door'* for $\lambda = 5$. The leftmost column are the initialised basis data matrices while the next two columns are the obtained data matrices after learning when all training data is used with a semantic balancing parameter of $\eta = 0$ and $\eta = 100$ respectively. The rightmost column are the obtained basis data matrices for $\eta = 100$ when only one training example per sound class is used. 74

4.6 Classification scores of LRM-NMD in single-label learning mode for $\eta = 100$ and $\lambda = 5$ together with the obtained baseline results in function of different number of training examples per sound class (n_{tr}). 76

4.7 Background noise robustness of LRM-NMD in single-label learning mode when $\eta = 100$ and $\lambda = 5$ in function the signal-to-noise ratio (SNR) for different number of training examples per sound class (n_{tr}). 77

4.8 Graphical visualisation of the definition $n_{overlap}$. Situation (a) corresponds to the situation $n_{overlap} = 0\%$ where the events are all separated by T frames. Situation (b) to $0\% < n_{overlap} < 100\%$ and the amount of overlapping frames is defined in percentage w.r.t. the shortest one. Situation (c) corresponds to $n_{overlap} = 100\%$ where all events have the same onset time. 78

4.9 Influence of the semantic balancing parameter (η) on the classification performance of LRM-NMD in low-resolution multi-label learning mode in function of different number of labelled events per segment (n_{lbl}) when $\lambda = 5$ and $n_{overlap} = 0\%$. 79

4.10 Classification scores of LRM-NMD in low-resolution multi-label learning mode on the non-overlapping data ($n_{overlap} = 0\%$) for $\eta = 5$ and $\lambda = 5$ together with the obtained baseline results in function of different number of labelled events per segment (n_{lbl}). 81

4.11 Background noise robustness of LRM-NMD in low-resolution multi-label learning mode when $\eta = 5$ and $\lambda = 5$ in function of the signal to noise ratio (SNR) for different number of labelled events per segment (n_{lbl}). 82

4.12 Classification scores of LRM-NMD in low-resolution multi-label learning mode for $\eta = 5$ and $\lambda = 5$ in function of the amount of overlap within the training events ($n_{overlap}$) on clean data for different number of labelled events per segment (n_{lbl}). 83

5.1 Graphical visualisation of the used system architecture. The architecture is composed of an acoustic front-end, an acoustic pre-processing stage and a local on-edge presence detection classifier. 91

5.2 Floor plan of the used office environment together with the desk setup and sensor locations. The acoustic sensor was placed next to the door entrance at an height of approximately 1.2 meters. The two small rooms indicated by '*Room A/B*' were not used during the recordings. 106

5.3 Graphical representation of the offline learning strategy. The relation between the amount of training data and presence detection performance is investigated by increasing the number of training examples offline and retraining the model parameters from scratch. Note that each batch has a balanced number of presence and absence samples. 109

5.4 The offline presence detection results for the three different time sequences. The obtained results are expressed in function of the evaluation metric (d) and the number of support vectors/prototypes (M). 111

5.5 The obtained results by the adaptive learning strategy using a fixed-model architecture in function of the training size and the number of prototypes (M) for the normal and reverse time sequence. 113

5.6 The influence of the threshold τ on the presence detection performance of SR-FSLSSVM for $M = 500$ and $M = 2000$ in the normal time sequence. 115

5.7 The obtained results by the adaptive learning strategy using an adaptive-model architecture in function of the training size and the number of prototypes (M) for the normal and reverse time sequence. The obtained results are expressed in function of the evaluation metric (d) and the number of support vectors/prototypes (M). 116

List of Tables

3.1 Overview of the recorded ADL dataset. 38

3.2 Processing time gain in the feature extraction module for the different feature parameter settings compared to the baseline setting: sampling frequency of 32 kHz, 20 MEL-filters and 14 cepstral coefficients. 43

3.3 Clean GMM classification results for the different feature parameter settings and the p-value of the paired student t-test for the best classification score of 8 kHz and 16 kHz against the highest classification score of 32 kHz. 47

3.4 Clean SVM classification results for the different feature parameter settings and the p-value of the paired student t-test for the best classification score of 8 kHz and 16 kHz against the highest classification score of 32 kHz. 47

3.5 Confusion matrix (in %) of the SVM approach for a sampling frequency of 16 kHz, 15 MEL-filters and 14 cepstral coefficients. The obtained classification performance is $78.0 \pm 2.8\%$. 48

3.6 Confusion matrix (in %) of the SVM approach when the best node number is included as feature for a sampling frequency of 16 kHz, 15 MEL-filters and 14 cepstral coefficients. The obtained classification performance is $81.1 \pm 2.3\%$. 49

4.1 Overview of the used NAR dataset. 67

4.2 Confusion matrix of LRM-NMD in single-label learning mode for $\eta = 100$ and $\lambda = 5$ when the number of training events per class is set to $n_{tr} = all$. 75

4.3 Confusion matrix of LRM-NMD in single-label learning mode for $\eta = 100$ and $\lambda = 5$ when the number of training events per class is set to $n_{tr} = 2$. 75

4.4 Confusion matrix of LRM-NMD in multi-label learning mode for $\eta = 5$ and $\lambda = 5$ when all events are labelled in each segment ($n_{lbl} = 5$). 80

4.5 Confusion matrix of LRM-NMD in multi-label learning mode for $\eta = 5$ and $\lambda = 5$ when two out of five events are labelled in each segment ($n_{lbl} = 2$). 80

5.1 Overview of the computational complexity and the required memory allocation of the pre-processing stage. 93

5.2 Overview of the recorded dataset. 106

5.3 Overview of the number of presence and absence $\mu\sigma$-MFCC feature frames per fold before balancing the training sets. . . . 107

5.4 The obtained tpr's and fpr's in the offline learning mode for the smallest, middle and largest batch size. 112

5.5 The obtained tpr's and fpr's for the adaptive learning mode using a fixed-model architecture after 1000, 13000 and 27000 training samples. 114

5.6 The obtained tpr's and fpr's for the adaptive learning strategy using an adaptive-model architecture after 1000, 13000 and 27000 training samples. 117

7.1 Classification scores of LRM-NMD in single-label learning mode on the clean data in function of the number of training examples per sound class (n_{tr}), semantic balancing parameter (η) and sparsity penalisation (λ). 136

7.3 Classification scores of LRM-NMD in low-resolution multi-label learning mode for $\eta = 5$ and $\lambda = 5$ in function of the number of labelled events per annotating segment (n_{lbl}), signal to noise ratio (SNR) and degree of overlap in the annotating segments ($n_{overlap}$). 137

7.2 Classification scores of LRM-NMD in low-resolution multi-label learning mode on the clean data in function of the number of labelled events per annotating segment (n_{lbl}), semantic balancing parameter (η) and sparsity penalisation (λ) 138

Chapter 1

Introduction

This chapter gives a brief overview of acoustic indoor monitoring and its applications in the current technological evolution. The observed trends and shortcomings regarding acoustic monitoring will be thoroughly discussed. This brings the introduction to the goals of this PhD thesis and its main research questions. Finally, a chapter by chapter overview of this dissertation will be given.

1.1 The era of Internet of Things and Big Data

The number of connected sensors and devices has increased exponentially over the last two decades as shown in Figure 1.1. A '*connected device*' is a device that uses electronics to capture data, process it and exchange it with other connected devices. This rapid increase is mainly a result of the decreasing hardware production costs and the fact that current off-the-shelf sensors are becoming cheaper, smaller and more energy efficient. As a result, devices and environments can be equipped with a wide set of sensors on a discreet level without significant additional costs to improve their user functionality. A well-known example is the breakthrough of smartphones in the mid 2000's which can be used for a series of other tasks besides calling and texting as well. For instance, current models are equipped with motion sensors (e.g. gyroscopes and accelerometers) which are typically used for automatically generating activity reports such as the number of footsteps, walked distance, time seated, etc. The ability of sensing the environment in combination with the facility to exchange the locally processed data among other sensors allow devices to operate interactively, which is to some extent autonomously resulting into a '*smart*' behaviour. Several well-known types of smart devices nowadays are smart watches, smart televisions, smart thermostats, smart speakers, etc.

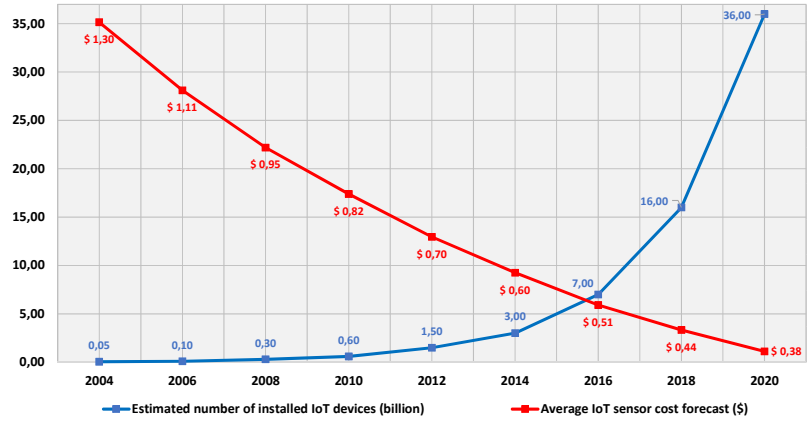

Figure 1.1: The estimated number of installed IoT devices vs. the average IoT sensor cost forecast in 2020 [1][153, 154].

[1]**Note:** The listed prices are based on the average selling prices for accelerometers/basic image sensors and not the average cost for a complete integration into a connected IoT device.

All these connected devices in our daily lives are known by the Internet of Things (IoT) or alternatively as the Internet of Everything (IoE). The term IoT was first introduced in 1999 by Kevin Asthon (brand manager at Procter & Gamble) during a presentation about the idea of using radio-frequency identification (RFID) tags in P&G's supply chain [4]. It is expected that the number of connected devices will reach up to 35 billion in 2020, and even up to 75 billion in 2025 (average estimates) [153, 154]. Connections made through IoT-enabled devices facilitate the rapid and efficient transfer of data needed to support a wide range of activities and operations. In this way, the application of IoT technologies can lead to significant operational improvements such as increased efficiency, performance and safety as illustrated in Figure 1.2 where the concept of a '*smart world*' is introduced [106]. For instance, video cameras beside roads can be used for regulating traffic flows in order to prevent traffic congestions [113, 171], environmental sensors can be used to monitor the air quality on-the-fly at different places and alert the public services in case of a pollution [50, 97], water quality sensors can be used to continuously monitor the city drinking water and alert the residents in case of a contamination [142, 61], (infrared) cameras can be used to adaptively control the street lights when pedestrians, (motor)cyclists or cars are detected at night or in low-visibility weather conditions and thereby saving energy [118, 172], and so on.

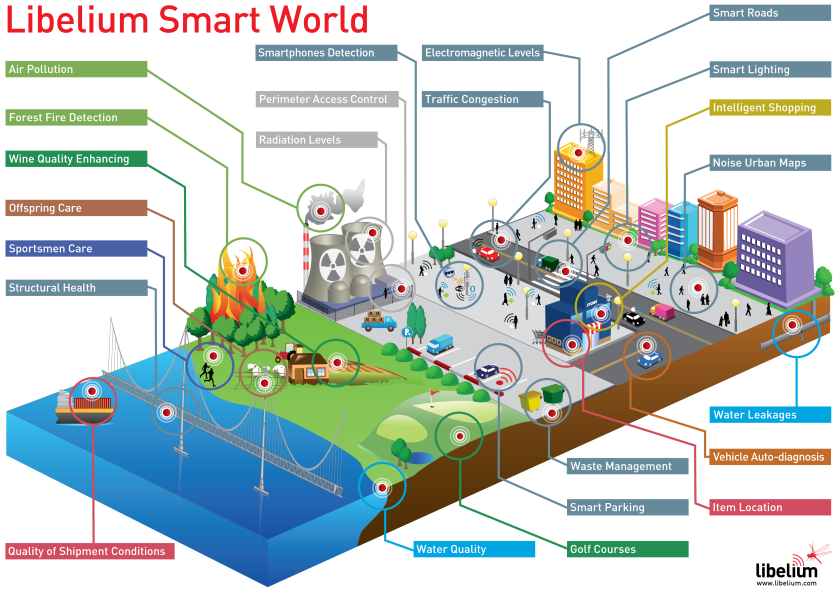

Figure 1.2: Conceptual visualisation of a smart city [106].

The consulting firm McKinsey and Company published in May 2018 a comprehensive study of the economic potential of IoT and its applications [110]. According to this study, the total economic benefit could hit between $3.9 trillion (3.9% GWP[2]) to $11.1 trillion (11% GWP) in 2025. The study also projects that industrial applications could see the greatest economic benefit of the IoT technology (situated between $1,210 billion and $3,700 billion) followed by cities ($930 billion and $1,660 billion) and health applications ($170 billion and $1,590 billion).

All these devices in the IoT network will produce enormous amounts of valuable sensor data. It is expected that the global amount of data on the internet will increase from 1.2 Zettabytes (1,200 trillion Gigabytes) in 2016 to 3.3 Zettabytes in 2021 [32]. Managing and analysing such enormous amounts of complex data cannot be done with the traditional database management software applications and is referred to as '*big data*'. Big data is commonly characterised in the literature by IBMs 5Vs model [94, 74]. The first and by far most important attribute is '*volume*' and refers to the quantity of generated and stored data. The amount of collected data determines the value and potential insight, and whether it can be considered as big data or not. The second attribute is '*velocity*' and refers to the speed at which the data is generated and processed. For instance, these days many applications require a real-time processing to meet the demands and challenges that lie in the path of growth and development. The third attribute is '*variety*' and defines the type and nature of the data, e.g. audio, video, text, etc., that must be stored and processed by the search algorithms. The fourth attribute is '*veracity*' and defines the uncertainty of the data. Being able to collect many different types of data coming in at a high rate makes it susceptible to faults or errors and thereby resulting in a poorer data quality. The final attribute is '*value*' and refers to the added value for companies. Having access to endless amounts of data is one thing, but it becomes all useless and even very expensive if it cannot be turned into an added value or profit for companies or the community.

Although these days vast amounts of data can be collected with a minimal financial cost, the main problem is often how to '*understand*' the information captured by the sensors. The lack of a correct data interpretation impedes the efficient processing of the information when searching for results or trying to extract useful relations in the source data. Therefore, a lot of effort must be devoted to the definition and the integration of semantics for an easier processing of the data by the search algorithms. In addition, many of today's applications also prefer a local (pre-)processing of the data at sensor side instead of cloud computing due to privacy considerations, restricted power consumptions or limitations in the communication bandwidth [162]. However, at the sensor

[2]Gross world product

side there are often limited computational resources available and stringent constraints in energy consumption. This results in the need of computational and memory efficient machine learning algorithms which can be mapped locally (i.e. on the gateway) and is known by '*machine learning on the edge*' [39, 164].

1.2 Relevance of acoustic indoor monitoring

The most straightforward approach for automatically monitoring indoor environments is the use of solely IoT-enabled devices which are already available on the market. For instance, the operating times of smart appliances can be used for analysing the daily living patterns of the residents [22] or to generate an alarm when the stove is left on when nobody is at home [7]. However, replacing all devices in an existing environment with their '*smart*' counterpart to achieve this objective is very expensive and ecological unfriendly [96]. This implies that such solutions are typically of less interest for already equipped environments. Another strategy is upgrading '*classical*' devices with after market communication facilities to obtain a smart behaviour as well. For instance, providing classical devices with RFID tags and IoT-enabled connections provide some level of smartness, however this requires many tags and could be aesthetically intrusive [78, 96]. Another approach is the use of general purpose sensors, e.g. microphones, (infrared) cameras, radars, etc., which can monitor a single room with only a limited number of sensors and without the need of direct instrumentation of objects.

Acoustic monitoring of both indoor and outdoor environments gained a lot of research interest in recent years. One of the main reasons for the increased interest is the improved computing power in the current off-the-shelf '*mobile*' hardware platforms enabling local acoustic processing of the data without the need of expensive mainframes. For instance, today's highest performance processor in the ARM Cortex mobile M-series, i.e. the ARM Cortex-M7, has a maximum clock frequency of 400 MHz and 2 MB internal program memory and is already available on the market between $10 and $20 depending on the casing and dependencies [124]. The ARM Cortex-M7 is especially developed for a wide variety of application areas including automotive, industrial automation, medical devices, high-end audio, image and voice processing, sensor fusion, advanced motor control and the deployment of IoT [124].

The use of acoustics for indoor monitoring applications has some interesting advantages compared to other sensing modalities. First of all, acoustic sensors can collect information from the environment at a distance and thus not requiring any physical contact with the objects being monitored. This implies that a

single acoustic sensor can be used to monitor multiple objects at once or even an entire room and thereby reducing the required number of sensors significantly. Secondly, acoustic data is highly informative and contains a lot of information about the environment being monitored. For instance, the spectral domain gives insight in the type of sound events that have been occurred, e.g. footsteps, speech, opening/closing doors, etc., while spatial filtering techniques allow to extract information about the location of the sound source [173]. Being able to gain insight about the origin of sound source is also beneficial for indoor monitoring applications since some sound event classes typically occur at a specific region in the environment, e.g. vacuum hood, toilet flush, etc. Thirdly, acoustic monitoring does not require a line-of-sight (LOS) path with the sound sources being monitored. This implies that acoustic observation systems are typically less sensitive to occlusions, i.e. obstructions in the LOS path, compared to video cameras or passive infrared (PIR) motion detectors. Finally, acoustic monitoring of indoor environments is also perceived as less privacy intrusive compared to video based monitoring in some specific situations. More specifically, it is safe to assume that end-users prefer acoustics over video based monitoring systems in rooms where they typically need to undress on a regular basis such as bathrooms, bedrooms and toilets. However, algorithmic considerations must be made to assure maximum privacy security. For instance, local featurisation and classification of the recorded sound signals preserves the essence within the data, but makes it typically impossible (depending on the types of features being used) to convert it back to its original state and thereby revealing what actually has been said in the environment in case speech was recorded.

Although the increased research interest in acoustic monitoring of environments, it is still an undervalued approach in the scientific community compared to other setups. A short overview of some interesting use cases, whether or not already investigated, are:

- **Home automation**: automatically selecting the pre-defined user specific profile of the home automation system, e.g. radio station, volume, mood lightning, temperature, etc., by doing voice or gait based person identification [136, 55, 56], or controlling the lights and appliances at home by means of vocal commands [2, 151, 3].
- **Health care**: continuously monitoring elderly at home allow caregivers to make an objective assessment of the health condition with a minimal personal effort. This relieves the increasing workload required from the nursing personnel as a result of the ageing population. For instance, frequently skipping meals or a significant increase in toilet visits might indicate an underlying health related issue [27, 53, 120]. In addition, the

monitoring system can also be used to automatically notify caregivers when an abnormality, e.g. fall incident, scream, etc., is detected. [119, 131, 132].

- **Security systems**: acoustically monitoring home environments is also an added value to conventional home security systems. The latter are typically based on PIR sensors and door/window contacts only. By including acoustics as modality allows the system to generate an alarm when the sound of an abnormality such as breaking glass or forcing a door is detected [122, 6]. In addition, the same security system can also be used to detect other abnormalities such as fire, water or gas leakage and thereby further improving the safety of the residents [166, 185, 156].

- **Industrial machinery analysis**: vibrations and sound signals produced by machinery are vital indicators of machine health and can be used to evaluate a machine's condition and the specific cause and location of problems. Early detection in machinery wear allows proactive maintenance scheduling and thereby reducing the risk of further damage. In addition, acoustic monitoring of machinery can be done while the machine is still operating. This results in lower down times and thereby improving the cost effectiveness of the machine being monitored [5, 104].

- **Facility services**: automatically generating status reports of the sanitary system (or other systems) in public buildings. These reports can be used by the facility service personnel for proactive maintenance planning, e.g. visual inspection of a toilet after each number of flushes or to refill the paper towel dispenser after a certain number of dispenses, or to generate an alarm in case of malfunctions, for instance when a urinal is making an abnormal sound or keeps flushing.

1.3 Current trends and their bottlenecks

This decade the scientific community of acoustic event and scene classification, which also covers the domain of acoustic indoor monitoring, underwent a shift from the traditional machine learning approaches to the extensive use of '*deep learning*' (DL) based architectures. This shift is clearly noticeable in the submissions of the '*IEEE Detection and Classification of Acoustic Scenes and Events*' (DCASE) challenges organised to catalyse the progress in the scientific community. Methods used in these challenges nicely list-up the state-of-the-art technologies in this field. The first DCASE challenge was organised in 2013 and had a total of 18 submissions[3] which were all based on a traditional machine

[3]Participants are allowed to submit different system outputs. Each system output is considered as a submission. Baselines are not considered as submissions.

learning approach [159]. The second DCASE challenge, organised in 2016, had a total of 82 submissions[2] and were equally balanced between traditional machine learning strategies and deep learning [176]. The most recent DCASE challenge was organised in 2018 and had a total of 234 submissions[2] and were practically all based on deep learning [130].

A '*deep neural network*' (DNN) is basically an artificial neural network (ANN) with multiple hidden layers between the input and output layer as shown in Figure 1.3 (a-b). The number of hidden layers defines the depth and complexity of the network. Each layer in a neural network is modelled by an architecture inspired by the biological neurons in the human brain. The very basic ingredient of any ANN are the so-called '*artificial neurons*' or nodes. An artificial neuron computes its output, often referred to as '*activation*', by mapping the weighted sum of its inputs optionally added with a bias to a non-linear activation function as shown in Figure 1.3 (c). The non-linearity of the activation function induces a non-linear behaviour in the neural network enabling modelling high-level abstractions in the data.

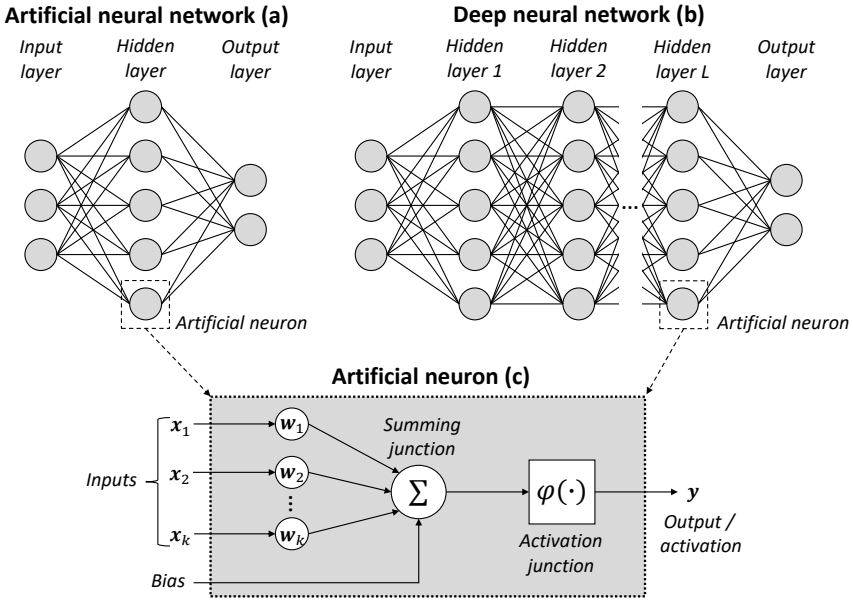

Figure 1.3: The difference between an ANN and DNN is situated in the number of hidden layers between input and output as shown in (a) and (b). The internal structure of an artificial neuron is shown in (c).

The rapid emergence and success of deep learning over the traditional machine learning techniques in recent years is mainly the result of three reasons. First of all, the architecture of a DNN is inspired by the neurons in the human brain and are very well-suited for modelling high-level abstractions in the data but are computationally expensive to solve [11, 144, 163]. Recent advances in computing power have enabled deep learning as a valid machine learning strategy for many different real-life applications and tasks. Secondly, the use of (leaky) rectified linear units (ReLu's) as activation function speeds up the learning capacity of the network by eliminating the so-called '*vanishing gradient*' problem [90, 109]. The vanishing gradient problem arises in DNNs that uses the default sigmoid curve or hyperbolic tangent as activation function whose gradients are situated in the range between 0 and 1. Since these small gradients are multiplied during backpropagation, they tend to "vanish" throughout the layers, preventing the network from learning dependencies in the data. Thirdly, deep learning also requires much more data to learn its model specific parameters compared to the traditional machine learning approaches [144, 163]. The rapid emerge of IoT-enabled devices facilitate the acquisition of vast amounts of data (i.e. big data) without the need of expensive and sophisticated hardware. As a result, the improved computing power in combination with the elimination of the vanishing gradient problem and the availability of vast amounts of data for training made deep learning the preferred machine learning strategy in the scientific community.

Although deep learning is a part of the broader family of conventional machine learning methods, there are some significant differences between both as shown in Figure 1.4. The overall objective of a machine learning task, and thus also of deep learning, is to enable machines to learn from the provided data and labelling support in order to make accurate predictions on newly collected samples. With traditional machine learning one must manually define the type of features being used to obtain a decorrelated representation of the data to train the predefined machine learning algorithm. The learned model then evaluates the extracted features of new input samples when analysing or classifying. Deep learning on the other hand typically skips the step of manually extracting the features resulting in some kind of black box behaviour [148]. More specifically, the data can be directly fed into the neural network, without explicitly defining information about the type of features or classifier being used[4], and the network is trained such that a given loss function is minimised [179]. This mechanism of learning results in some kind of automatic feature extraction since each level in the network transforms the input data into slightly more abstract and composite

[4]Note: although deep learning is more robust to correlated input data, the use of a decorrelated data representation can typically achieve the same performance but with a less complex model architecture. Hence, in acoustic processing it is common to use a spectral representation of the data as input data.

representations which are known as feature maps [99]. For example, let us take a deep learning image processing and classification task. The first set of hidden layers in the DNN will typically be trained to model the individual pixels, the second set of hidden layers will be used to model edges while the last set of hidden layers will model specific shapes. The final classification result of the input image is produced in the output layer.

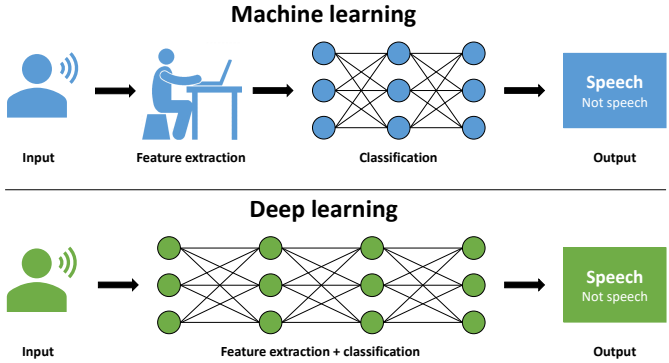

Figure 1.4: The conceptual difference between a traditional machine learning task and deep learning. Traditional machine learning requires a manual selection of the feature extraction and classification algorithm being used. Deep learning merges both stages in an automatic fashion without the need of explicitly defining them.

There are already many different forms of DNNs described in the literature, e.g. convolutional neural networks (CNN) [100, 80], recurrent neural networks (RNN) [138, 80], etc., each being optimised for a specific task. Despite the great success of deep learning, it comes at the cost of high computational complexity resulting in the need of expensive mainframes or cloud computing for training the networks. As a result, one of the aspects that gained a lot of research interest in the scientific community is situated in reducing the computational complexity of the machine learning algorithms. Using less complex learning strategies speeds up the training time and relaxes the need of expensive servers and mainframes. In addition, some applications require a local embedded processing of the data at sensor side because of multiple reasons such as privacy considerations, energy constraints or limited communication bandwidths [162]. For instance, domestic image and audio processing applications typically prefer a local recognition over cloud processing due to privacy and data protection aspects [47, 147], wearable medical devices for monitoring physiological signals of patients are

battery powered resulting in the need of a resource and energy efficient data processing for an increased autonomy [30, 175, 103], etc. Despite the increased computing power in today's mobile devices (e.g. ARM Cortex-M7), it still typically lacks sufficient resources for embedding complete state-of-the-art deep learning approaches locally. Traditional machine learning can offer a solution due to their reduced complexity compared to deep learning. Manually selecting the type of features and classifier in advance for a specific pre-defined task enables optimised implementations which are more convenient to be embedded locally [162]. Pushing machine learning from the cloud or gateway (edge computing) to the sensor will be called in this work by '*machine learning on the extreme edge*' and is also gaining research interest in the scientific community.

Another aspect that gained a lot of research interest is the ability of the machine learning algorithms to cope with '*weak supervision*' labelling support since providing strong labels to all data in today's big data era is simply an impossible task. Strong labels are an exact description of what has been occurred in the data. For instance, in case of acoustic signal processing the occurrence of sound events such as footsteps, running faucet, coughing, etc. Having access to the labels enables machine learning to extract relations within the data which can be used later for processing new unseen data. Weak supervision on the other hand is defined by high level and less accurate (noisy) labels which might not entirely correspond to the ground truth of the data but which are significantly cheaper to acquire. In general there are three types of weak supervision and are categorised as '*incomplete*', '*inexact*' and '*inaccurate*' supervision [187]. Incomplete supervision is the situation when not all events in the data are labelled as occurring. Inexact supervision (or tagging) is the situation when the labelling information does not contain accurate time content, i.e. a larger segment is supported with labelling information simply indicating the (subset of) events that have been occurred. Inaccurate labelling is the situation when not all labels correspond to the ground truth situation, i.e. annotation errors. Although weak labels typically not entirely correspond to the ground truth due to the annotation errors, it still contain valuable information for the machine learning algorithms to improve their ability to discover relations in the data. In addition, the ability of the machine learning algorithm to cope with overlapping sound events is also an important aspect. An acoustic scene is basically a sequence of potentially overlapping sound events, e.g. footsteps in combination with speech, which may result in different acoustical properties regarding their isolated counterpart. Algorithmic design choices must be made in order to deal with these changed properties due to the overlap as well.

Another important aspect is the plasticity of the embedded machine learning algorithms. It is desirable that the learned models can be adapted to different applications or environments (e.g update of the weights and model in the

classifier) without the need of a complete retraining procedure from scratch requiring all past seen samples. Different strategies are already investigated in the literature and are known by '*transfer learning*'. Transfer learning is the application of knowledge gained from completing one task to help solve a different but related problem [128, 127]. For instance, models learned to recognise cats could be applied as starting point to recognise dogs. Another important aspect is the ability of the machine learning algorithm to continuously update its model parameters to the actual situation in a stepwise manner from small batches of newly arrived samples only. The latter is known by '*online*' or '*adaptive*' learning and reduces the required memory footprint of the embedded platform significantly [129, 105, 62]. For instance, let's take again the task of recognising dogs. When a new unseen breed occurs in the dataset, the model parameters must be adapted to represent the newly collected information as well. Although transfer learning and online/adaptive learning are already widely examined in the scientific community, the ability for mapping these solutions to devices with limited resources is still highly underinvestigated.

1.4 Aim of this research

The aim of this doctoral research is the use of acoustics as sensing modality for indoor monitoring applications. The main focus is situated in the use of '*machine learning on the extreme edge*', i.e. energy and resource efficient algorithms which can be embedded on the current off-the-shelf mobile hardware platforms and endpoint IoT-enabled devices (Challenge I). As already discussed in Section 1.1, the emergence of IoT-enabled devices has led to a big data era where large amounts of data can be collected from the environment without the need of expensive acquisition hardware. Extracting relationships from the sensor data requires a correct data interpretation, however providing strong supervision support to all data is practically impossible due to the high cost of the labelling process. As a result, it is desirable that the machine learning algorithms can cope with the available '*weak supervision*' support such that an improved interpretation of the data can be obtained (Challenge II). The ability of the machine learning algorithm to learn its model parameters directly from data containing overlapping sound events will also be investigated since it often occurs in a real-life setting as well (Challenge III). Furthermore, the relations in the data might also vary over time, e.g. change in daily living patterns due to seasonal changes or an underlying health condition. This implies that the learned models must also adapt over time to the actual situation (Challenge IV). This brings in the '*adaptive learning*' aspect where models are updated if needed from the acoustic data labelled by user specific corrections (e.g. manually

controlling the lights or setting the heat to the preferred level) or external sensor labelling information (e.g. PIR based presence detection).

To summarise, the three main research contributions of this doctoral dissertation in the field of acoustic indoor monitoring are:

- **Supervised classification of activities of daily living from acoustic sensor data** (Challenge I): This part investigates two simple and resource efficient supervised machine learning classifiers, i.e. a support vector machine (SVM) and a Gaussian mixture model (GMM), regarding classification accuracy and computational complexity in runtime mode. Algorithmic trade-offs are made in the signal processing and machine learning part to enable local or '*on the edge*' processing of the data on the current off-the-shelf mobile hardware platforms. Real-life data of activities of daily living (ADL) was recorded using a wireless acoustic sensor network (WASN). Such networks have the advantage that no inconvenient cabling is required between the sensors (reduces installation effort and improves user acceptance) and can easily be scaled with additional microphones to achieve a higher spatial sampling. Furthermore, the influence of domestic background noise on the classification performance of the WASN is also examined and compared to single-microphone solutions, i.e. network of microphones per room vs. one microphone per room.
- **Weakly supervised convolutive acoustic event modelling and classification** (Challenge II & III): This research topic focusses on the development of a convolutive learning algorithm capable of modelling acoustic events from weakly supervised data. The weak supervision is implemented on a '*low-resolution*' and '*multi-label*' level (LRM). More specifically, LRM labelling simply indicates the (subset of) events that take place in a larger acoustic segment (e.g. 30 seconds of data) without defining any of the actual onsets and offsets. The convolutive event modelling is done by means of a non-negative matrix deconvolution (NMD). The objective of NMD is to factorise an acoustic stream into the convolution between a set of fundamental dictionary items, i.e. the primary sound events (e.g. cough, hand clap, opening/closing door, etc.), and their activation pattern over time such that the reconstruction error is minimised. The non-negativity constraint of NMD results in an additive behaviour of the model, i.e. an acoustic stream is represented as a linear combination of isolated sound events, making that it can directly deal with overlapping sound events in the data without any modifications. However, the main disadvantage of NMD is that the factorisation occurs in a blind fashion, i.e. without any labelling support, which may converge to sub-optimal solutions. This work proposes an extension to NMD where

the factorisation is done such that the reconstruction error on both the acoustic data and the available weakly supervised labelling information is minimised and is called LRM-NMD. A publicly available dataset of isolated domestic events was used to evaluate LRM-NMD and to compare it with other state-of-the-art machine learning algorithms. In addition, the influence of overlapping events in the training set and the robustness to environmental background noise are examined as well.

- **On-edge adaptive acoustic presence detection** (Challenge I & IV): The contribution of this topic is situated in the development of an acoustic presence detection algorithm which can be embedded locally on devices with limited computational resources. The focus of the classifier is situated in discriminating '*presence*' from '*absence*' related sound data, i.e. a binary classification task, and not in classifying individual sound events. Local processing of the data, i.e. feature extraction and classification, with limited resources is known by '*on-edge*' computing and has the advantage that no sensitive data needs to be transmitted resulting in an enhanced user privacy. The '*adaptive*' aspect yields that the model parameters are continuously updated from the newly collected samples and the corresponding labelling information. In this work a least-squares support vector machine (LS-SVM) based learning strategy will be used. LS-SVM has as advantage over a traditional SVM setup that optimality is defined by a set of linear equations instead of a quadratic programming (QP) problem. Solving such systems can easily be done on devices with limited computational resources using already available optimised microcontroller unit (MCU) libraries. However, standard LS-SVM formulations are known to not result in a sparse solution and thus becoming very expensive to solve in case of large-scale datasets. In this work two approximation schemes are investigated to enforce a sparse solution and are known by '*fixed-size*' LS-SVM (FS-LSSVM) and '*subset-of-regressors*' LS-SVM (SR-LSSVM). The adaptive learning aspect is introduced by updating the model parameters from the incoming training data using a recursive least-squares (RLS) implementation. A real-life dataset was collected over a period of 28 days resulting in 555 hours of data. All data was labelled on-the-fly by means of a simple button press when entering or leaving the room. Two different adaptive learning strategies are proposed, i.e. one with a '*fixed-model*' architecture (i.e. fixed set of prototype vectors) and one with an '*adaptive model*' architecture (i.e. set of prototypes varies over time), and are investigated for the purpose of acoustic presence detection.

1.5 Chapter by chapter overview

This dissertation is organised in six chapters. The next chapter discusses the main components of a supervised acoustic classification algorithm. The following three chapters present our own contributions made in the field of acoustic indoor monitoring. The general discussion, conclusions and future research plans are given in the last chapter. A short chapter by chapter overview is given next:

- **Chapter 2** introduces the main components of a supervised classification algorithm. The aspects of training and testing are introduced and how the 'best' model parameters are learned from a set of labelled data. In addition, the acoustic feature extraction process and the baseline classification algorithms used in this research are also explained in this chapter.
- **Chapter 3** studies ways to obtain a computationally efficient classification architecture to classify activities of daily living from data recorded by a WASN in the context of elderly health care. Automatically classifying ADLs allows to generate summary reports of the performed activities over a longer period of time. These reports can be used by the caregivers to evaluate the health condition and to assess the self-reliance of the older person in a more objective manner. A dataset of ADLs was recorded in a real-life home environment using a WASN as acquisition system. The collected data was labelled manually on the activity level instead of the event level to speed up the annotation process. The focus of this research is situated in the use of simple and resource efficient machine learning classifiers which can be mapped to the current off-the-shelf endpoint IoT-enabled devices. Furthermore, the experiments in this chapter also investigate the computational complexity of the proposed system architecture in runtime mode and the robustness to background noise.
  - Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Automatic monitoring of activities of daily living based on real-life acoustic sensor data: A preliminary study." In proceedings of 4th workshop on speech and language processing for assistive technologies, SLPAT 2013, Grenoble, France, Aug. 2013, pp. 113-118.
  - Vuegen, L., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Automatic monitoring of activities of daily living using a wireless acoustic sensor network." In proceedings of 5th Dutch Bio-Medical Engineering Conference 2015, BME 2015, Egmond aan Zee, The Netherlands, Jan. 2015.

  - Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Monitoring activities of daily living using wireless acoustic sensor networks in clean and noisy conditions." In proceedings of 37th international conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2015, Milan, Italy, Aug. 2015, pp. 4966-4969.
  - Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Energy efficient monitoring of activities of daily living using a wireless acoustic sensor network in clean and noisy conditions." In proceedings of 23rd European Signal Processing Conference, EUSIPCO 2015, Nice, France, Sept. 2015, pp. 449-453.
  - Luca, S., Vuegen, L., Van hamme, H., Karsmakers, P., and Vanrumste, B. "Decision support systems for home monitoring applications: Classification of activities of daily living and epileptic seizures.", Bookchapter: In Institution of Engineering and Technology, IET 2016, London, UK, pp. 271-291.

- **Chapter 4** proposes a weakly supervised learning strategy based on non-negative matrix deconvolution (NMD). The labelling information is directly included to the optimisation problem of NMD on a low-resolution and multi-label (LRM) level. The LRM labelling simply indicates the events, or even a subset of the events, that take place in a larger acoustic segment (e.g. 30 seconds) without indicating the onsets and offsets. This approach in labelling the data speeds up the annotation process significantly and reduces the human effort in labelling the data. A publicly available acoustic dataset of domestic sound events was used to evaluate LRM-NMD and to compare it with other state-of-the-art classification algorithms. In addition, the experiments in this chapter also investigate the influence of overlapping sound events in the training data and the robustness to environmental background noise.

  - Vuegen, L., Karsmakers, P., Vanrumste, B., and Van hamme, H. "Acoustic event classification using low-resolution multi-label non-negative matrix deconvolution." In Journal Audio Engineering Society, AES vol. 66, nr. 5, May 2018, pp. 369-384.
    (**Awarded as the best JAES paper of 2018**)
  - Vuegen, L., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Weakly supervised classification of domestic acoustic events for indoor monitoring applications." In proceedings of IEEE Conference on Biomedical and Health Informatics 2018, BHI 2018, Las Vegas, USA, March 2018.
  - Meire, M., Vuegen, L., and Karsmakers, P. "The impact of missing labels and overlapping sound events on multi-label multi-instance

learning for sound event classification." Accepted for the proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events, DCASE 2019, New York, United-States, Oct. 2019.

- **Chapter 5** discusses an on-edge adaptive acoustic presence detection strategy using a least-squares support vector machine (LS-SVM) framework. Two different adaptive learning strategies are proposed in this chapter, i.e. one with a fixed-model architecture and one with an adaptive-model architecture, to update the model parameters and are evaluated to a baseline setting where all model parameters are estimated offline. A real-life acoustic dataset was recorded at an office environment over a period of 28 days. The data was labelled by the users by a simple button press when entering or leaving. The focus of the experiments is situated in investigating the relation between the number of prototypes (linked to the computational complexity of the algorithm) and the amount of seen training data regarding presence detection performance.

  – Vuegen, L. and Karsmakers, P. "On-edge adaptive acoustic presence detection." In final VLAIO report for the O&O project: Research towards a perceptive sensing home control system (PERSES).

- **Chapter 6** reviews the most important results of this doctoral dissertation and discusses some interesting future research directions.

Chapter 2

Main aspects of supervised acoustic classification

This chapter discusses the main ingredients of a supervised acoustic classification algorithm. The aspects of training and testing are introduced and how the 'best' model parameters are chosen. Next, the acoustic feature extraction process will be explained together with the baseline classification algorithms used in the experiments of this thesis.

2.1 Introduction

The main objective of a classification task in machine learning is to find a function that maps the instances or examples from some input space $\mathcal{X}$, defined by the set of attributes or features, to the output space $\mathcal{Y}$. The input space $\mathcal{X}$ can be seen as a set of all possible instances, whether they are present in the dataset or not, and the output space $\mathcal{Y}$ is defined by the class names. Generally, the classification task t can thus be seen as $t : \mathcal{X} \rightarrow \mathcal{Y}$ in T.

For further clarification of the notations above, let us introduce following toy problem: the objective is to classify incoming e-mails whether they are *'spam'* or *'not spam'* from the two words occurring the most in the body of the corresponding e-mail. Hence, each e-mail $\mathbf{x}$ is represented in the input space by $\mathbf{x} = \{x_1; x_2\} \in \mathcal{X}$, with x_1 and x_2 the so-called attributes or features, and the output space, also known as label space, is given by $\mathcal{Y} \in \{spam; not\ spam\}$. For instance, it can be assumed that an e-mail with features $\mathbf{x} = \{lottery; winner\}$ is more likely to be *'spam'* than *'not spam'* and that the e-mail $\mathbf{x}' = \{appointment; doctor\}$ can be considered as *'not spam'*.

We, in terms of frequent e-mail users, can make the assumptions above by ourselves due to the knowledge which we have learned over time by using e-mail. The same procedure must be performed by the machine learning classification task, i.e. the algorithm needs to learn what the typical features are regarding *'spam'* and *'not spam'*, in order to perform the task $t : \mathbf{x} \in \mathcal{X} \rightarrow \mathcal{Y}$ as well as possible. This procedure is called *learning* or *training* and requires a set of labelled examples in case of supervised learning.

Let us denote the training set by $X^{TR} = \{(\mathbf{x}_1; y_1), (\mathbf{x}_2; y_2), \ldots, (\mathbf{x}_N; y_N)\}$ with $y_n \in \mathcal{Y}$ as the true label of example $\mathbf{x}_n$. Thus, the objective is to learn the model specific parameters from the training set such that the classification task is performed in a 'best' possible way given some performance metric. This leads to an important aspect in machine learning namely *generalisation*. Generalisation is defined as the ability of the model, learned from the finite set of N training examples, to generalise to future unseen data as well. For instance, too much focus on modelling the tiniest detail in the training set, by using a complex classification algorithm with a high number of model parameters, typically yields good scores on the training data given the performance metric but leads to poor generalisation, i.e. the model is not representing to unseen future data. This phenomenon is called *overfitting*. Conversely, *underfitting* occurs when the classification algorithm is not able to model the training data correctly which also leads to insufficient generalisation.

Figure 2.1 is the block scheme of a baseline supervised classification algorithm. As one can see, the raw training set is transformed to the input space by means

of the feature extraction algorithm. The set of training features in combination with the true class labels are then used to learn the model specific parameters. Once the learning procedure is finished, the model is then ready for predicting unseen test samples.

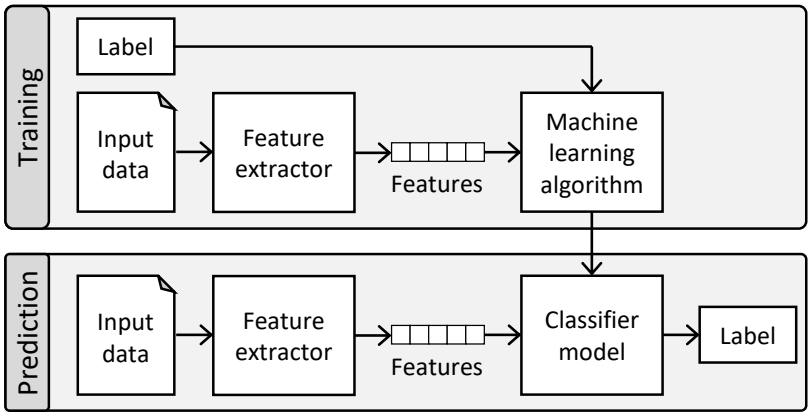

Figure 2.1: Block scheme of a standard supervised machine learning task. The upper part visualises the training aspect of the classifier while the lower part corresponds to the prediction phase.

The remainder of this chapter is structured as follows. First, in Section 2.2 we will discuss the acoustic features used in this doctoral research and how they are extracted from the acoustic data. Secondly, in Section 2.3 we will thoroughly explain the baseline classification algorithms used in the experiments of this thesis and how the model parameters are learned during training.

2.2 From raw data to acoustic features

Transforming the acoustic data into so-called acoustic feature vectors is an important stage in the classification process. As already mentioned, the overall goal of a classification task is to distinguish between examples of different classes, in general the goal of feature extraction is to reduce the variability of features from examples that are associated with the same class, while increasing the distinction between features from examples that belong to different classes [59, 85].

Hence, an ideal acoustic feature:

- has large interclass variabilities and a small intraclass variability.
- is robust against noise and distortion.
- will not be affected by small long-term variations.
- occurs frequently and naturally in the acoustic data.
- is easy to extract from the acoustic data.

Many different acoustic feature extraction algorithms have been proposed in the literature over the recent years and can be categorised into two main categories which are referred to as *temporal* and *spectral* features [59]. The temporal features, computed directly from the acoustic data in the time domain, are simple to extract and have an easy physical interpretation but are typically insufficient for complex classification tasks. Examples of temporal features are short-term energy of the signal, zero crossing rate, maximum amplitude, etc. The spectral features on the other hand, which are computed in the frequency domain, are more descriptive and are thus generally considered as the core features of a sound classification system. In this thesis, the MEL-scale filter bank spectral analysis is used as main feature extraction algorithm during the experiments and will be thoroughly explained in the next section.

2.2.1 MEL-frequency Cepstral Coefficients

The MEL-frequency cepstral coefficients (MFCC) feature extraction algorithm finds its origin in the domain of speech/speaker recognition and is based on the perception of sound by the human auditory system [35]. Despite the fact that MFCCs are mainly developed for speech and speaker related tasks, it is shown that they are also usable for non-speech related tasks as well such as acoustic event or scene classification for instance [159, 176]. Hence, the MFCCs are the dominant features used for speech and audio classification tasks and their success is due to their ability to represent the amplitude spectrum in a compact and decorrelated form.

Figure 2.2 is an overview of the MFCC feature extraction process and involves the following main components: (1) framing and windowing, (2) discrete Fourier transform, (3) MEL-frequency warping and (4) cepstral analysis. Each step in the process of creating MFCC features is motivated by perceptual or computational considerations.

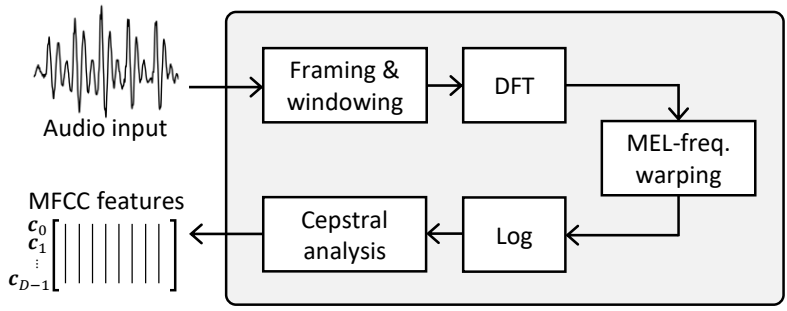

Figure 2.2: MEL-frequency cepstral coefficients feature extraction process. The raw acoustic data is transformed to the feature domain by applying a (1) framing and windowing, (2) discrete Fourier transform, (3) MEL-frequency warping and (4) cepstral analysis on the log-transformed MEL-magnitude features.

2.2.1.1 Framing and windowing

The framing and windowing operation of the feature extraction process transfers the raw acoustic waveform into short overlapping segments. These segments, further called frames, are typically 20 to 30 ms long with an overlap of more than 50%. Within such a window, it is reasonable to assume that audio data is wide-sense stationary. Each frame is then windowed with an Hamming-window to prevent leakage during the spectral analysis. The Hamming-window is defined by

$$\mathbf{h}_n = 0.54 - 0.46 \cos\Big(\frac{2\pi n}{N-1}\Big), \tag{2.1}$$

with N denoting the number of samples in each frame and $n = [0, 1, \ldots, N-1]$. The windowing operation of a frame $\mathbf{f}$ is thus defined by $\mathbf{s}_n = \mathbf{h}_n \mathbf{f}_n$.

2.2.1.2 Discrete Fourier transform

The framing and windowing operation are followed by the discrete Fourier transformation (DFT) where the windowed frames are transformed from the time domain into the frequency domain by using

$$\mathbf{x}_k = \sum_{n=0}^{N-1} \mathbf{s}_n e^{-j(\frac{2\pi n k}{N})}, \tag{2.2}$$

with $k = [0, 1, \ldots, N-1]$ and $\mathbf{x}_k$ containing the frequency content of bin k. Typically, the frames are zero-padded to the next radix-2 number such that

the fast Fourier transformation (FFT) can be used which is a computational efficient variant for computing the DFT [33].

Next, only the magnitude spectrum $|\mathbf{x}_k|^2, k = [0, 1, \ldots, N-1]$, is retained from the spectral information since perceptual studies of the auditory system have shown that the amplitude of the spectrum is of more importance than the phase information [59].

2.2.1.3 MEL-frequency warping

The MEL-frequency warping smooths the high-dimensional magnitude spectrum and transforms it into perceptually meaningful frequencies. More specific, the MEL-scale reflects the sensitivity of the human auditory system to frequency shifts. Although one would expect that frequency shifts are perceived equally in the frequency range, it has been found that the lower frequencies are perceptually of more importance than the higher ones [157]. For instance, a frequency shift of Δf at 1 kHz is perceived stronger than the same shift at 10 kHz. Therefore, the MEL-frequency scale follows a mapping such that a shift of Δf_{MEL} is perceived equally regardless its location in the frequency spectrum and is defined by

$$f_{MEL} = 2595 \log_{10}(1 + \frac{f}{700}), \tag{2.3}$$

with f the original frequency (in Hertz) and f_{MEL} the perceived counterpart (in MEL).

One approach to simulate the subjective spectrum is by using the so-called MEL-scale filter bank which is defined by overlapping triangular frequency response bandpass filters with a constant spacing and bandwidth in the MEL-frequency domain as shown in Figure 2.3.

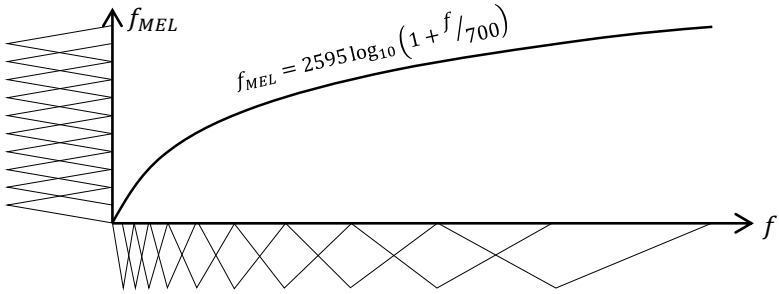

Figure 2.3: The relation between the perceptual MEL-frequency scale (f_{MEL}) and the traditional frequency scale (f). The MEL-filter bank spanning 10 bands are uniformly distributed at the MEL-warped spectrum, or non-uniformly at the original frequency spectrum.

The MEL-magnitude features can now be defined as the energies observed in each band. Typically, the number of bands (B) is set in the range between 20 and 40 [108, 183]. In order to use these bands, the inverse of Equation 2.3 is used to calculate the weights of each band at all N frequency components. Hence, the MEL-magnitude features can be computed by using:

$$\mathbf{m}_b = \sum_{k=0}^{N-1} \mathbf{W}_{bk} |\mathbf{x}_k|^2, \tag{2.4}$$

with $b = [0, 1, \ldots, B-1]$ and $\mathbf{W}_{bk}$ the weight of the MEL-scale filter bank in band b at frequency k.

2.2.1.4 Cepstral analysis

The last stage in the computation of the MFCC feature vector converts the log-transformed MEL-magnitude features to the pseudo time domain, also known as *'quefrency'*, by using a discrete cosine transform (DCT). The DCT on the log MEL-magnitude features is defined by

$$\mathbf{c}_d = \sum_{b=0}^{B-1} (\log \mathbf{m}_b) \cos\left[d\left(b + \frac{1}{2}\right)\frac{\pi}{B}\right], \tag{2.5}$$

with $d = [0, 1, \ldots, D-1]$ and $[\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{D-1}]$ as the D-dimensional MFCC feature vector. Typically the number of cepstral coefficients (D) in the domain of speech and audio processing is set to 14 and is thus lower than the number of

MEL-bands [108, 183]. In addition, the DCT decorrelates the highly-correlated MEL-magnitude features which is beneficial for the acoustic modelling later on.

2.2.1.5 Delta and delta-delta

Optionally, the MFCC feature vectors are stacked with their first and second order derivations to augment the features with some temporal context. The first order derivation ($\Delta\mathbf{c}$) models the change between two successive feature while the second order derivation ($\Delta\Delta\mathbf{c}$) models the rate of change.

2.3 Baseline acoustic classifiers

As already mentioned in the introduction of this chapter, the objective of a classification task in machine learning is to map instances from the input space $\mathcal{X}$ to the label space $\mathcal{Y}$ by using $t : \mathcal{X} \rightarrow \mathcal{Y}$. Throughout the years many different approaches have been proposed in the literature to achieve this goal for different tasks. In this thesis, we will use Gaussian mixture models (GMM) and support vector machines (SVM) as baselines since they are already widely examined in the domain of acoustic signal processing [160, 176].

As a reminder, the following notation will be maintained during the remainder of this chapter unless explicitly stated otherwise. The D-dimensional input space $\mathcal{X} \in \mathbb{R}^D$ is represented by the set of N feature vectors using the notation $\mathbf{x}_n = \{x_{n1}, x_{n2}, \ldots, x_{nD}\} \in \mathcal{X}$, with $n = [1, 2, \ldots, N]$, and their class label $y_n \in \mathcal{Y}$.

2.3.1 Gaussian mixture model

A Gaussian mixture model is a weighted sum of M-multivariate Gaussian distributions modelling the input space [12]. The density function $p(\mathbf{x})$ of a GMM is defined by

$$p(\mathbf{x}) = \sum_{j=1}^{M} w_j b_j(\mathbf{x}), \tag{2.6}$$

where w_j are mixture weights that satisfy the constraint $\sum_{j=1}^{M} w_j = 1$ and $b_j(\mathbf{x})$, $j = [1, 2, \ldots, M]$, are the M density functions of the D-dimensional multivariate Gaussian distributions. The multivariate Gaussian density function is given by:

$$b_j(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_j|^{1/2}} exp\Big(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1}(\mathbf{x} - \boldsymbol{\mu}_j)\Big), \tag{2.7}$$

with mean vector $\boldsymbol{\mu}_j$ and covariance matrix $\boldsymbol{\Sigma}_j$. The complete set of GMM parameters are collectively represented by $\lambda = \{w_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}$, $j = [1, 2, \ldots, M]$, and can be estimated from the set of observed features $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ by maximising the likelihood (ML) function which is given by

$$p(\mathbf{X}|\lambda) = \prod_{n=1}^{N} \Big[\sum_{j=1}^{M} w_j b_j(\mathbf{x}_n) \Big]. \tag{2.8}$$

Due to the summation over j in Equation 2.8, the maximisation is not analytically tractable inducing the need for a numerical approach as the expectation-maximisation (EM) algorithm [37]. Basically, EM starts with an initial model λ, to estimate the new model $\hat{\lambda}$, such that $p(\mathbf{X}|\hat{\lambda}) \geq p(\mathbf{X}|\lambda)$. The new model becomes the initial model for the next iteration and the process is repeated until some convergence threshold is reached or the maximum number of iterations is exceeded. The corresponding update formulas are

$$w_j = \frac{1}{N} \sum_{n=1}^{N} p(j|\mathbf{x}_n, \lambda), \tag{2.9}$$

for mixture weight w_j,

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} p(j|\mathbf{x}_n, \lambda)\mathbf{x}_n}{\sum_{n=1}^{N} p(j|\mathbf{x}_n, \lambda)}, \tag{2.10}$$

for mean vector $\boldsymbol{\mu}_j$ and

$$\boldsymbol{\Sigma}_j = \frac{\sum_{n=1}^{N} p(j|\mathbf{x}_n, \lambda)(\mathbf{x}_n - \boldsymbol{\mu}_j)(\mathbf{x}_n - \boldsymbol{\mu}_j)^\top}{\sum_{n=1}^{N} p(j|\mathbf{x}_n, \lambda)}, \tag{2.11}$$

for the covariance matrix $\boldsymbol{\Sigma}_j$ with

$$p(j|\mathbf{x}_n, \lambda) = \frac{w_j b_j(\mathbf{x}_n)}{\sum_{i=1}^{M} w_i b_i(\mathbf{x}_n)}. \tag{2.12}$$

A graphical visualisation of a 2D input space modelled by a GMM containing two mixture components is given by Figure 2.4. As one can see, the two dense regions of data points in the input space, located at $\{\mu_{11}; \mu_{21}\}$ and $\{\mu_{12}; \mu_{22}\}$ respectively, are each modelled by one mixture component. The total probability distribution $p(\mathbf{X}|\lambda)$ of the mixture model is shown by the contour lines.

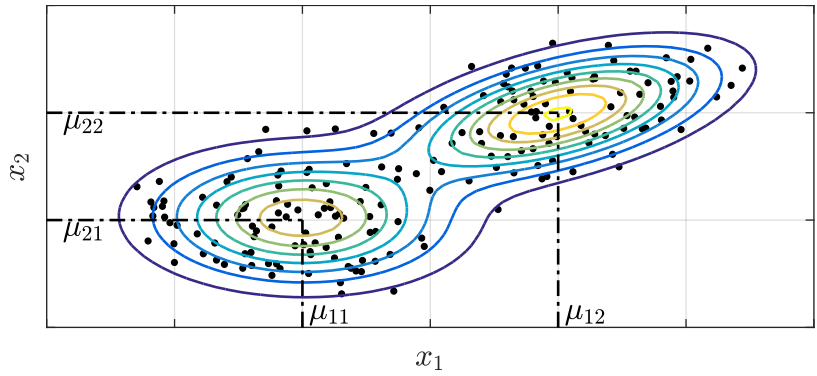

Figure 2.4: Graphical visualisation of a 2D input space modelled by a Gaussian mixture model containing two mixture components.

For classification, the group of C classes are represented by their set of GMM parameters $\{\lambda_1, \lambda_2, \ldots, \lambda_C\}$ and the objective is to find the class model which has the maximum-a-posteriori (MAP) estimate given a test feature vector $\bar{\mathbf{x}}$. The MAP-estimate is defined by

$$\hat{c} = \underset{1 \leq c \leq C}{\operatorname{argmax}}\, p(\lambda_c|\bar{\mathbf{x}}) = \underset{1 \leq c \leq C}{\operatorname{argmax}}\, \frac{p(\bar{\mathbf{x}}|\lambda_c)p(\lambda_c)}{p(\bar{\mathbf{x}})}, \tag{2.13}$$

where the second equation is obtained by using Bayes' theorem [12]. The terms in Equation 2.13 are:

- $p(\lambda_c|\bar{\mathbf{x}})$ is the probability that the correct class is λ_c given the test feature vector $\bar{\mathbf{x}}$. This is called the *posterior* or *a posteriori* probability since it can only be estimated after the observation of $\bar{\mathbf{x}}$.
- $p(\bar{\mathbf{x}}|\lambda_c)$ is the density function of $\bar{\mathbf{x}}$ given the model parameters λ_c, also referred to as the *likelihood* function, and is computed by using Equation 2.6.
- $p(\lambda_c)$ is the *prior* or *a priori* probability of class c since it can be determined before the observation of $\bar{\mathbf{x}}$.
- $p(\bar{\mathbf{x}})$ is the probability of the test feature vector $\bar{\mathbf{x}}$, which is called the evidence.

Hence, Bayes' rule expresses the *posterior* probability in terms of both the likelihood function and the prior probability. Typically, Equation 2.13 is further

reduced into

$$\hat{c} = \underset{1 \leq c \leq C}{\operatorname{argmax}}\, p(\bar{\mathbf{x}}|\lambda_c)p(\lambda_c), \tag{2.14}$$

due to the class independence of $p(\bar{\mathbf{x}})$. This implies that the test feature vector $\bar{\mathbf{x}}$ is classified to the class which obtains the highest product of likelihood and prior probability.

2.3.2 Support vector machine

A support vector machine is fundamentally a binary classifier that constructs a decision function in the input space which separates the positive instances from the negative ones. Let us recall the training set of N examples $(\mathbf{x}_n, y_n)$ where each $\mathbf{x}_n \in \mathbb{R}^D$ is a vector in the D-dimensional input space and $y_n \in \{-1, +1\}$ is its corresponding class label. The main idea of a SVM is to construct the decision function $f(\mathbf{x})$, called *hyperplane*, in such way that the two sets of examples are separated and that the distance of $f(\mathbf{x})$ to the closest examples, i.e the *support vectors*, is maximised [16].

The reasoning behind maximising the distance originates from the assumption that examples situated close together in the input space are likely to have the same classification, and therefore it makes sense to prefer a decision function $f(\mathbf{x})$ that stays as far away from the training examples. The distance between the nearest support vectors from both classes to the hyperplane is called the *margin* and needs to be maximised. Therefore, SVMs are also known in the literature as *maximum margin classifiers*.

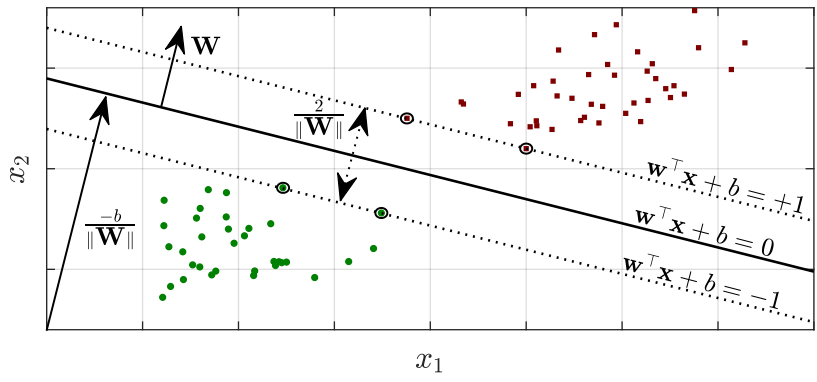

Figure 2.5: Illustration of the margin of a support vector machine in case of linearly separable data. The maximum margin is defined only by the support vectors which are marked by a circle.

A hyperplane in the D-dimensional input space can be defined by a set of points satisfying $\mathbf{w}^\top \mathbf{x} + b = 0$ where $\mathbf{w}$ is the normal vector to the hyperplane and $\frac{-b}{\|\mathbf{w}\|}$ determines the offset from the origin along $\mathbf{w}$. The objective of a SVM is to estimate $\mathbf{w}$ and b such that $y_n(\mathbf{w}^\top \mathbf{x}_n + b) \geq 1$, $n = [1, 2, \ldots, N]$, and that the distance from the decision hyperplane to the support vectors is maximised. The margin is defined by the distance between the planes parallel to the decision function that lie on the positive and negative support vectors, i.e. $\mathbf{w}^\top \mathbf{x} + b = +1$ for the positive class and $\mathbf{w}^\top \mathbf{x} + b = -1$ for the negative class respectively. The margin can be geometrically expressed by $\frac{2}{\|\mathbf{w}\|}$, so maximising the margin yields minimising $\|\mathbf{w}\|$. This leads to following optimisation problem

$$\mathbf{w}^*, b^* = \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2, \tag{2.15}$$

subject to (s.t.) $y_n(\mathbf{w}^\top \mathbf{x}_n + b) \geq 1$, $n = [1, 2, \ldots, N]$, and is called the *primal* learning problem of a *hard*-margin SVM. Since perfectly separable classes in real-life data are fairly uncommon, the constraint in Equation 2.15 cannot be satisfied for all training examples and thus needs to be relaxed such that it admits a solution for overlapping classes as well. The idea is to introduce *slack* variables ξ_n, one for each example, which allows *margin* errors, i.e. training examples inside the margin or even at the wrong side of the decision boundary. The slack variables can be categorised in three classes namely $\xi = 0$ for the training examples on the correct side of the decision boundary, $0 < \xi \leq 1$ for the training examples in the margin and $\xi > 1$ for the training examples on the wrong side of the decision boundary. Hence, the relaxed optimisation problem is defined by

$$\mathbf{w}^*, b^* = \min_{\mathbf{w}, b, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{n=1}^{N} \xi_n, \tag{2.16}$$

s.t. $y_n(\mathbf{w}^\top \mathbf{x}_n + b) \geq 1 - \xi_n$ and $\xi_n \geq 0$, $\forall n = [1, 2, \ldots, N]$, which is known as the *soft*-margin primal learning problem. The parameter C is the *trade-off* or *regularisation* parameter between margin maximisation and slack variable minimisation: a high value of C implies that margin errors incur a high penalty, while a low value permits more margin errors in order to achieve a large margin. The trade-off parameter C is also known in the literature as a complexity parameter since it controls to some extent the complexity of the SVM model, i.e. penalising margin errors, by increasing C, typically yields smaller margins modelled by a high number of support vectors.

The soft-margin primal learning problem given by Equation 2.16 can be expressed using the method of Lagrange multipliers. Adding the constraints into the

optimisation problem yields

$$\begin{aligned}\mathcal{L}(\mathbf{w}, b, \xi_n; \alpha_n, \beta_n) =& \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}\beta_n\xi_n \\ &- \sum_{n=1}^{N}\alpha_n(y_n[\mathbf{w}^\top\mathbf{x}_n + b] - 1 + \xi_n),\end{aligned} \tag{2.17}$$

with $\alpha_n \geq 0$ and $\beta_n \geq 0$ as the Lagrange multipliers. The Lagrangian can be solved by computing the partial derivatives $\frac{\partial\mathcal{L}}{\partial\mathbf{w}}$, $\frac{\partial\mathcal{L}}{\partial b}$ and $\frac{\partial\mathcal{L}}{\partial\xi_n}$ and setting them to zero. Rewriting the solution in terms of α_n leads to following convex quadratic constrained optimisation problem

$$\alpha_1^*, \ldots, \alpha_N^* = \max_{\alpha_1,\ldots,\alpha_N} -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m \mathbf{x}_n^\top\mathbf{x}_m + \sum_{n=1}^{N}\alpha_n, \tag{2.18}$$

s.t. $\sum_{n=1}^{N}\alpha_n y_n = 0$ and $0 \leq \alpha_n \leq C$, $n = [1, 2, \ldots, N]$. Equation 2.18 is known in the literature as the soft-margin *dual* learning problem and has some interesting properties. First, it shows that searching for the maximum-margin decision boundary is equivalent to searching for the support vectors, e.g. the training examples with non-zero Lagrange multipliers α_n. Secondly, it shows that the optimisation problem is entirely defined by pairwise dot products between the training instances which paves the way for a very powerful adaptation that allows to learn non-linear decision boundaries as well.

The main strategy in SVM to deal with non-linear decision functions is by transforming the data non-linearly to a higher dimensional *feature* space $\mathcal{V}$, using the mapping function $\varphi : \mathcal{X} \to \mathcal{V}$, in which linear classification can be applied. The remarkable thing of the pairwise dot products in Equation 2.18 is that the feature space $\mathcal{V}$ does not have to be explicitly constructed but that all necessary operations can be implicitly computed in the input space $\mathcal{X}$ which is known as the *kernel trick*. The kernel trick is defined by $k(\mathbf{x}_n, \mathbf{x}_m) = \varphi(\mathbf{x}_n)^\top\varphi(\mathbf{x}_m)$ and can be directly plugged into Equation 2.18 which yields

$$\alpha_1^*, \ldots, \alpha_N^* = \max_{\alpha_1,\ldots,\alpha_N} -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m k(\mathbf{x}_n, \mathbf{x}_m) + \sum_{n=1}^{N}\alpha_n. \tag{2.19}$$

A commonly used kernel is the so-called Radial Basis Function (RBF) which is defined by $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2/2\sigma^2)$ with σ^2 denoting the kernel bandwidth and needs to be tuned during training.

Once the SVM is learned, an unseen test sample $\bar{\mathbf{x}}$ is evaluated using

$$\hat{c} = \text{sign}\big[\sum_{n=1}^{N} y_n \alpha_n k(\bar{\mathbf{x}}, \mathbf{x}_n) + b\big]. \tag{2.20}$$

As already mentioned, a SVM is basically a binary classifier although many different techniques are proposed in the literature to deal with multi-class problems. The two most commonly used strategies are the so-called one-vs-all (1vA) and one-vs-one (1v1) approach. The 1vA topology involves the division of a C class dataset into a subset of C two-class cases while the 1v1 topology constructs a SVM for each pair of classes resulting into $(1/2)C(C-1)$ two-class cases. The overall classification output can then be computed by applying a majority voting over the sub-classification results. We have chosen to use the 1v1 topology in the experiments of this work because of the following two reasons. First of all, research has shown that the 1v1 topology is less sensitive to unbalanced training datasets compared to 1vA [63]. Secondly, although 1v1 constructs more classifiers, it is typically faster in learning since it keeps the size of the training dataset rather small compared to 1vA since only two classes are evaluated in each sub-model.

Chapter 3

Supervised models for classifying activities of daily living

This chapter is devoted to supervised classification of activities of daily living (ADL) from data recorded by a wireless acoustic sensor network (WASN). The computational complexity of the proposed system architecture in runtime mode will be investigated and how the optimal configuration parameters are chosen regarding algorithmic complexity and classification performance. In addition, the influence of background noise on the classification performance of the WASN is also examined in this chapter and compared to single-microphone solutions.

The work presented in this chapter was published in:

- Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Automatic monitoring of activities of daily living based on real-life acoustic sensor data: A preliminary study." In proceedings of 4th workshop on speech and language processing for assistive technologies, SLPAT 2013, Grenoble, France, Aug. 2013, pp. 113-118.
- Vuegen, L., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Automatic monitoring of activities of daily living using a wireless acoustic sensor network." In proceedings of 5th Dutch Bio-Medical Engineering Conference 2015, BME 2015, Egmond aan Zee, The Netherlands, Jan. 2015.

- Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Monitoring activities of daily living using wireless acoustic sensor networks in clean and noisy conditions." In proceedings of 37th international conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2015, Milan, Italy, Aug. 2015, pp. 4966-4969.

- Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Energy efficient monitoring of activities of daily living using a wireless acoustic sensor network in clean and noisy conditions." In proceedings of 23rd European Signal Processing Conference, EUSIPCO 2015, Nice, France, Sept. 2015, pp. 449-453.

- Luca, S., Vuegen, L., Van hamme, H., Karsmakers, P., and Vanrumste, B. "Decision support systems for home monitoring applications: Classification of activities of daily living and epileptic seizures.", Bookchapter: In Institution of Engineering and Technology, IET 2016, London, UK, pp. 271-291.

3.1 Introduction

Acoustic monitoring of elderly for health care related applications gained a lot of research interest in recent years. Demographic studies predict that the degree of eldery (older than 65 years) in the European population will increase from 20.5% in 2020 to 28.4% in 2060 [48, 49]. This ageing brings important challenges to our society to prevent rocketing costs in the elderly care sector. One of these challenges is to assist elderly people to stay as long and safe as possible in their own home environment with minimal personal assistance. This relieves the fast growing demand nowadays for expensive elderly care facilities.

Currently, the golden standard to determine the self-reliance of an elderly is the so-called Katz index of independence in activities of daily living (ADL), often referred to as the Katz ADL scale [83]. This scale measures the self-reliance by observing how well the following basic tasks are performed by the elderly without any assistance: *'bathing'*, *'dressing'*, *'toileting'*, *'transferring'*, *'continence'* and *'feeding'*. The major drawbacks of this approach in evaluating the self-reliance are the time and effort required from caregivers and that it is not always an objective evaluation since it is only a snapshot assessment of the person in question.

The aim of this research is to automatically compile a summary report over a longer period of time about the performed ADLs which can be used by caregivers to evaluate the self-reliance of an older person in a more objective manner. For instance, increasing activity durations over time or frequently skipping a meal might indicate a decreased self-reliance as a result of an underlying health issue. These reports will be generated by classifying domestic sounds acquired by a wireless acoustic sensor network (WASN) and linking them to one of the ADLs. The use of a WASN for this application has multiple advantages compared to other setups. For instance, the current off-the-shelf acoustic sensors are rather small, energy efficient and cheap, making them easy to integrate in the home environment on a discreet level with minimal costs. In addition, multiple microphones can be placed in the home environment achieving a large spatial sampling without any inconvenient cabling among the sensor locations [10]. This makes that WASN solutions typically have a higher degree of acceptance by the end users compared to wired solutions [111]. Furthermore, providing the acoustic sensors in the WASN with local processing power makes that the workload can be distributed among the sensors so that relatively inexpensive hardware can be used [10]. Hence, algorithmic considerations must be made such that the required algorithms can be implemented on an embedded platform with limited resources.

The remainder of this chapter is structured as follows. In Section 3.2 we discuss

the used acquisition system and the recorded ADL dataset. The proposed system architecture is discussed in Section 3.3 together with the computational complexity of the local feature extraction and the acoustic classifier in runtime mode. Section 3.5 introduces the simulation environment used to examine the noise robustness of the WASN. The obtained results in both clean and noisy conditions are thoroughly discussed in Section 3.6 while the concluding remarks are given in Section 3.7.

3.2 Collected dataset

3.2.1 Recording system

As already mentioned in the introduction of this chapter, we have opted to use a wireless acoustic sensor network as acquisition system to collect ADL data. The WASN used in this work consists of multiple so-called acoustic nodes as proposed in [173]. Each node in the sensor network is equipped with three linearly spaced MEMS microphones (SPU0410LR5H) with an inter-microphone distance of 2.5 cm. The SPU0410LR5H is a miniature, high-performance, low power microphone and is very well suited for both audio and ultrasound applications [87]. A single-ended amplifier, with RF/EMI protection and a gain-factor of 25.1 dB as advised in the datasheet, was used for pre-amplification of the sensor signals [88]. All captured acoustic signals were recorded using a four channel 24-bit sound card (ESI MAYA44 XTe) sampling at 96 kHz [44]. The synchronisation among the sensor nodes is achieved by transmitting an additional RF-synchronisation signal which is recorded by all acoustic nodes in the WASN as well. For a detailed overview of the complete wireless acoustic sensor network, the readers are advised to consult [173].

In addition, since each acoustic node in the WASN is equipped with a linear array of three microphones, spatial information can be extracted from the acoustic data as well. For instance, the sound source locations can be estimated or the nodes can be steered to listen at a specific region in the home environment. This is an interesting property of the WASN that might improve the classification performance of the wireless acoustic sensor network in certain circumstances (e.g. noisy conditions). However, it is important to note that the focus in this research is situated at utilising the spectral representation of the data only and that no spatial information will be extracted from the acoustic sensor data. More information about using spatial features for ADL classification can be found in [173].

3.2.2 Home environment

Figure 3.1 is the floor map of the home environment used for the recording of acoustic ADL data. In total seven different nodes, each marked by a red rectangular box with an arrow indicating the orientation of the node, were installed in the home environment at a height of approximately 1.75 m. Four of the seven nodes were placed at each corner of the combined living room and kitchen (nr. 1-4). The remaining three nodes were installed in the bedroom (nr. 5), toilet (nr. 6) and bathroom (nr. 7) respectively. This implies that each room of the home environment was covered during the experiments.

The noise source locations to examine the influence of background noise on the performance of the WASN are indicated by the blue stars A and B respectively. These positions are chosen such that each noise source is located at approximately 35 cm to one of the nodes in the WASN. The latter is done to examine if the large spatial sampling of the WASN yields better classification accuracies in noisy situations compared to single microphone solutions. For instance, it can be assumed that nodes further away from a noise source are less affected by the noise compared to the ones situated nearby. Utilising the acoustic data of the nodes further away might result into less contaminated features and thereby improving the classification performance of the WASN.

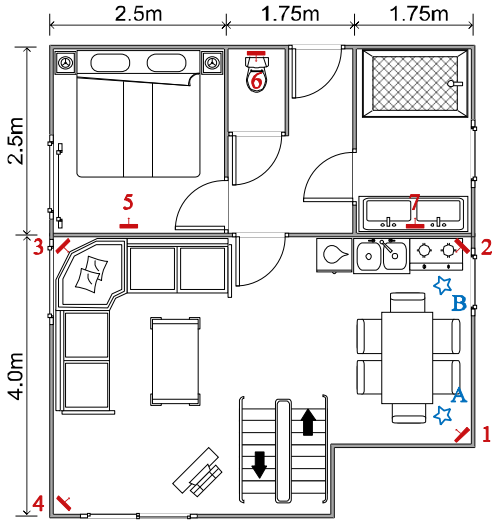

Figure 3.1: Floor plan of the home environment together with the node numbers (denoted by 1 to 7) and the two noise source locations (marked by A and B).

3.2.3 Recorded dataset

In total ten different activities of daily living were performed multiple times in the home environment by two different test users. All data was labelled manually on the activity level with the help of video data which has been recorded solely for annotation purposes. A detailed overview of the collected ADL data is given in Table 3.1.

Table 3.1: Overview of the recorded ADL dataset.

Cat.	ID	Class names	# Examples	Average duration (in min.)
Hygiene	1	Brushing teeth	6	2.0±0.2
	2	Showering	6	2.6±0.6
	3	Toileting	12	1.2±0.1
	4	Washing hands	6	0.9±0.1
Eating	5	Consuming meal	10	4.9±1.2
	6	Preparing food	4	8.7±1.7
	7	Setting table	6	4.8±1.0
	8	Washing dishes	4	7.0±0.6
Other	9	Dressing	24	1.5±0.6
	10	Sleeping	6	3.1±0.4

All eating related activities were performed in the combined living environment and kitchen. The personal hygiene related activities were performed in the bathroom with exception of toileting. Furthermore, sleeping was always recorded in the bedroom while dressing occurs both in the bedroom and bathroom.

3.3 System architecture

The proposed system architecture for ADL monitoring is shown in Figure 3.2 and consists of multiple acoustic nodes in combination with a central processing unit (CPU) for classification. In order to reduce the amount of data transfer between the nodes and the central processing unit, only the acoustically relevant features of the node receiving signal at the highest signal-to-noise ratio (SNR) will be computed and transmitted to the CPU. These ‘best’ features are then

used to learn the model specific parameters of the classifier during training and for evaluation in runtime mode.

The acoustic information in the WASN will be processed in blocks of 30 seconds which corresponds to the minimum activity duration in the observed ADL dataset. Each block is further partitioned into frames of 25 ms with a frame shift of 10 ms. A frame is either generated by a sound source related to an ADL or by a background noise source. For each 30 second block an average SNR is computed as the ratio between the average energy in the interesting frames and that in the noise related frames. Furthermore, only a single microphone channel is used in each node since no spatial information will be extracted from the acoustic data.

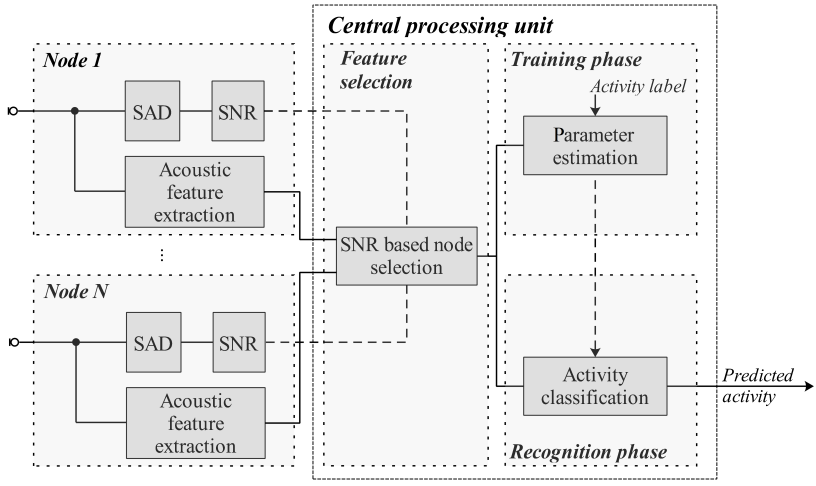

Figure 3.2: Overview of the proposed system architecture for ADL classification. Each node in the WASN determines the SNR of the received data and only the node with the highest SNR will extract the ADL relevant acoustic features. These 'best' features are then sent to the CPU to learn the model specific parameters of the classifier during training or for classification in runtime mode.

3.3.1 Sound activity detection and signal-to-noise ratio estimation

The objective of the sound activity detector (SAD) is to determine the acoustically relevant frames in each 30 second block captured by the sensor network. Since a wide range of acoustic events (each having its own

characteristics) are useful for ADL classification, a model specific SAD cannot be used which is typically tailored to detect a specific class of sounds. Therefore, a simple and computationally efficient energy based sound activity detector with an adaptive noise floor was implemented at each node in the WASN. Only the acoustically relevant frames will be transformed by the feature extraction module into the MFCC features.

The sound activity detector starts with estimating the energy in the received 25 ms frames by using

$$E_j = \sum_{n=1}^{N} |\mathbf{f}_{j,n}|^2, \tag{3.1}$$

where $\mathbf{f}_{j,n}$ denotes the nth sample in frame j. Next, the short-term energy S_j and noise floor N_j are estimated from E_j by means of

$$S_j = (1 - \alpha^{(s)})S_{j-1} + \alpha^{(s)}E_j, \tag{3.2}$$

and

$$N_j = \begin{cases} (1 - \alpha^{(s)})N_{j-1} + \alpha^{(s)}E_j & \text{if } N_{j-1} \geq S_j, \\ (1 - \alpha^{(l)})N_{j-1} + \alpha^{(l)}E_j & \text{if } N_{j-1} < S_j, \end{cases} \tag{3.3}$$

respectively. These formulas are first order exponential smoothing functions with α as smoothing factor, and satisfies following condition $0 \leq \alpha^{(l)} \leq \alpha^{(s)} \leq 1$. The relation between the smoothing factor α and the time constant τ, i.e. time for a unit set function to reach 63.2% of the original signal, is expressed by $\alpha = 1 - e^{\frac{\Delta T}{\tau}}$ with ΔT as time interval between two frames [168].

This implies if there is sufficient energy in the captured sound signal, i.e. when $N_{j-1} < S_j$, the noise floor N_j will be updated slowly with $\alpha^{(l)}$ and remains quasi unchanged. On the other hand, if there is a low amount of energy in the recorded sound signal, i.e. when $N_{j-1} \geq S_j$, the noise floor N_j will be updated rapidly with $\alpha^{(s)}$ and thereby following the energy level of the background noise. For every frame j in the 30 second block, the corresponding SAD-flag D_j is defined by

$$D_j = \begin{cases} 1 & \text{if } Ej > N_j + \epsilon, \\ 1 & \text{if } E_j \leq N_j + \epsilon \text{ and hangover is not expired}, \\ 0 & \text{if } E_j \leq N_j + \epsilon \text{ and hangover is expired}, \end{cases} \tag{3.4}$$

where ϵ is an additional margin to prevent toggling of the SAD-flag during long periods of silence. The hangover on the other hand avoids clearing the SAD-flag during suddenly energy drops in the acoustic data and adds a short amount of silence frames after each detection as well. All SAD configuration parameters, i.e. $\alpha^{(l)}$, $\alpha^{(s)}$, ϵ and hangover timer, were tuned manually per node during the

installation of the WASN in the home environment. The smoothing factors $\alpha^{(s)}$ and $\alpha^{(l)}$ where chosen such that the SAD rapidly detects sound in order to prevent missing events and are $\alpha^{(s)} = 1/5$ and $\alpha^{(l)} = 1/50.000$ corresponding to a time constant of 50 ms and 500 seconds respectively given a ΔT of 10 ms (frame shift). The hangover timer was set for 1 second and the additional margin ϵ was chosen node specifically in function of the stationary sensor noise.

The signal-to-noise ratio (SNR) is defined by

$$SNR_{dB} = 10 \log_{10} \left(\frac{Es}{En} \right), \tag{3.5}$$

where Es and En corresponds to the average energy of the SAD detected frames (acoustically relevant) and the non-SAD detected frames (background noise) in each 30 second block respectively. Two special cases are when there are no frames of either type in a given 30 second block. In case of only acoustically relevant frames, the average background energy of the previous 30 second block is used to compute the SNR. In case of only background frames, the corresponding 30 second block will be omitted since it doesn't contain any acoustically relevant ADL data.

3.4 Acoustic feature extraction

As already mentioned in the introduction of this chapter, algorithmic considerations must be made regarding the feature extraction process such that these can be computed locally on an embedded platform with limited resources. Hence, the computational complexity of the MFCC feature extraction process is examined in terms of the required number of multiplications defined by $\mathcal{O}$, also known in the literature as the big-O notation, in function of the different feature configuration settings.

The extraction of a MFCC feature vector involves a framing and windowing operation, a discrete Fourier transformation, a MEL-frequency warping and a cepstral analysis as discussed in Section 2.2.1. Each step in the process of the feature extraction requires a specific amount of multiplications and can be defined as follows.

1. **Windowing**: each frame is windowed with an Hamming-function to prevent leakage due to the segmentation. This operation requires $\mathcal{O}(N)$ multiplications with N denoting the number of samples in each frame. The number of samples on the other hand depends linearly on both the frame length and the sampling frequency fs.

2. **Discrete Fourier transformation**: the windowed frames are transformed into the frequency domain by using the fast Fourier transformation (FFT) which is a computational efficient implementation of the DFT. The total cost of the FFT is expressed by $\mathcal{O}\big(K \log_2(K)\big)$ with K denoting the number of frequency bins. Due to the symmetrical property of the DFT and FFT, only the lowest $K/2$ frequency bins need to be retained for the MEL-frequency warping.

   Important to note is that the number of frequency bins for the FFT must be a radix-2 number. As a result, each windowed frame is zero padded to the next radix-2 number making $K \geq N$.

3. **MEL-frequency warping**: the frequency spectrum of each frame is filtered by a set of MEL-filters and the resulting power of each filter is calculated. This MEL-frequency warping can be computed by a matrix multiplication which results in a total of $\mathcal{O}(BK')$ multiplications with B the number of MEL-filters in the filterbank and K' denoting the lowest $K/2$ frequency bins. However, the computational cost of $\mathcal{O}(BK')$ is in fact a slight overestimation since most of the elements in the filterbank are zeros and can be omitted during the matrix multiplication.

4. **Cepstral analysis**: the frequency cepstrum is computed by applying a discrete cosine transformation (DCT) on the log-transformed MEL-features. This is done by multiplying the log-transformed MEL-features with a DCT-matrix. As a result, the computational cost is defined by $O(CB)$ with C indicating the number of cepstral coefficients.

Table 3.2 gives an overview of processing time gain for different parameter settings compared to the baseline setting, i.e. a sampling frequency of 32 kHz, 20 MEL-filters and 14 cepstral coefficients. This baseline setting requires in total 21,560 multiplications for computing a single feature vector for a window size of 25 ms. As one can see, the most important parameter to reduce the computational cost is the sampling frequency followed by the number of MEL-filters. Reducing the number of cepstral coefficients yields only a slight processing time gain at the feature extraction module, however it significantly reduces the required bandwidth of the wireless communication between the local nodes and the central processing unit. In addition, reducing the number of cepstral coefficients is also beneficial for the computational complexity of the acoustic modelling and classification process later on. Furthermore, the computational complexity of the MFCC approach can also be further reduced by using an optimised extraction strategy as discussed in [69]. However, in this research the default MFCC extraction algorithm is used as discussed in Section 2.2.1.

Table 3.2: Processing time gain in the feature extraction module for the different feature parameter settings compared to the baseline setting: sampling frequency of 32 kHz, 20 MEL-filters and 14 cepstral coefficients.

# MEL-filters	# cepstral coefficients	Sampling frequency 8 kHz	16 kHz	32 kHz
10	7	5.99	2.82	1.33
15	7	5.05	2.41	1.15
15	14	4.92	2.38	1.14
20	7	4.36	2.10	1.01
20	14	4.24	2.07	1.00

Note: The processing time gain is expressed as a fraction, e.g. a processing time gain of two yields two times faster computations.

3.4.1 Acoustic classification

During the experiments of this chapter we have examined a Gaussian mixture model (GMM) and a support vector machine (SVM) as an acoustic classifier for the purpose of supervised ADL classification. The computational complexity of both classifiers in runtime mode is also evaluated and discussed below. The cost of learning the model specific parameters on the other hand is not examined since it is assumed that these are estimated off-line. Furthermore, it is worth to note that the computational complexity of the classifier itself is of less importance compared to cost of the acoustic feature extraction algorithm since the classification will be done centrally on a CPU with sufficient processing power.

1. **Gaussian mixture models**: each activity in the ADL dataset is represented by a set of GMM parameters and the objective is to find the class model λ_c with the highest maximum-a-posteriori (MAP) score given the test vector $\bar{\mathbf{x}}$. The simplified MAP estimator is defined by

$$\hat{c} = \underset{1 \leq c \leq C}{\operatorname{argmax}}\, p(\bar{\mathbf{x}}|\lambda_c), \tag{3.6}$$

due to the class independence of $p(\bar{\mathbf{x}})$ and the assumption of equal class prior probabilities $p(\lambda_c)$ in the dataset. The a-posteriori probability $p(\bar{\mathbf{x}}|\lambda)$ is calculated by using

$$p(\bar{\mathbf{x}}|\lambda) = \sum_{j=1}^{M} \frac{w_j}{(2\pi)^{d/2}|\mathbf{\Sigma}_j|^{1/2}} exp\Big(-\frac{1}{2}(\bar{\mathbf{x}} - \boldsymbol{\mu}_j)^\top \mathbf{\Sigma}_j^{-1}(\bar{\mathbf{x}} - \boldsymbol{\mu}_j)\Big), \tag{3.7}$$

with M representing the number of mixture components in the GMM. Both the normalisation term $\frac{w_j}{(2\pi)^{d/2}|\mathbf{\Sigma}_j|^{1/2}}$ and the inverse covariance matrix $\mathbf{\Sigma}_j^{-1}$ are independent of $\bar{\mathbf{x}}$ and can thus be seen as a constant since they must be computed once for each mixture component. As a result, only the term $-\frac{1}{2}(\bar{\mathbf{x}}-\boldsymbol{\mu}_j)^\top\mathbf{\Sigma}_j^{-1}(\bar{\mathbf{x}}-\boldsymbol{\mu}_j)$ must be evaluated for each test vector $\bar{\mathbf{x}}$. This operation requires $\mathcal{O}(D^2+D)\approx\mathcal{O}(D^2)$ multiplications in case of a full covariance matrix and $\mathcal{O}(2D)\approx\mathcal{O}(D)$ multiplications when diagonal covariance matrices are used. In order to determine the MAP probability, Equation 3.7 must be performed C times. This implies that the total computational cost of a GMM classifier during evaluation is expressed by $\mathcal{O}(CMD^2)$ or $\mathcal{O}(CMD)$ depending on the type of covariance matrices used, i.e. full or diagonal, to evaluate a single test vector.

2. **Support vector machine**: during classification, an unlabelled test vector $\bar{\mathbf{x}}$ is evaluated by the two-class SVM model using

$$\hat{c}=\text{sign}\big[\sum_{n=1}^{N}y_n\alpha_n k(\bar{\mathbf{x}},\mathbf{x}_n)+b\big], \tag{3.8}$$

where $\mathbf{x}_n$, y_n and α_n, $n=[1,2,\ldots,N]$, are the model parameters with N denoting the total number of training features and $k(\bar{\mathbf{x}},\mathbf{x}_n)$ is the kernel function describing the similarity between two feature vectors. In this work we have opted to use an RBF kernel which requires $\mathcal{O}(ND)$ multiplications to evaluate $\bar{\mathbf{x}}$ against all $\mathbf{x}_n$. Since in total $\frac{1}{2}C(C-1)$ SVMs are required (one-vs-one topology) to expand this two-class classification problem into a multiclass variant, each having a set of $\frac{2N}{C}$ training examples, the overall computational cost of the complete SVM topology becomes $\mathcal{O}((C-1)ND)$ to classify a single test vector $\bar{\mathbf{x}}$. However, it is worth to note that only a limited number of $\mathbf{x}_n$ have a non-zero weight α_n, i.e. those points closest to the decision hyperplane. The latter implies that the given computational complexity is in fact an overestimation since the training examples having a zero α_n can be omitted.

3.5 Background noise simulation environment

A simulation environment was used to create an artificial noise dataset which can be combined with the clean ADL data in order to examine the influence of background noise on the classification performance of the WASN. The used simulation environment estimates the room impulse responses (RIR) from a particular noise source location to each microphone in the WASN on basis of the T_{60} time, the room dimensions and the microphone positions and orientations

using the image source method [65]. All configuration parameters were measured during the installation of the WASN in the home environment to parametrise the simulation model.

For the experiments in this chapter we have opted to use the publicly available CHiME dataset which contains clean examples of typical domestic noise sources such as speech, television and radio [8]. This noise dataset can be filtered by the obtained RIRs to generate an artificial background noise dataset. In total seven different SNR levels are examined, going from 30 dB to 0 dB in steps of 5 dB, for the two background noise source positions marked by the blue stars A and B respectively in Figure 3.1.

3.6 Experiments and results

3.6.1 Experimental setup

All experiments in this chapter were performed using a two-fold cross-validation two times which results in four equally sized training and test sets. The data in each fold was normalised such that the training set has zero mean and unit standard deviation in each feature dimension.

During the experiments, the maximum-likelihood parameters of the GMM models are estimated using the expectation-maximisation algorithm on the training data as discussed in Section 2.3.1. Research indicates that GMMs with ten mixture components are typically sufficient for the classification of activities of daily living [177]. Hence, each GMM in the experiments of this chapter is represented with ten mixture components modelled by full covariance matrices.

The SVM hyperparameters on the other hand are selected by applying a cross-validation in the training dataset. In this research a radial basis function was used as kernel. This implies that during each fold a grid search over the trade-off parameter C and the kernel parameter σ is performed to find their optimal value. In contrast to GMM, SVM typically uses the normalised mean and standard deviation of each cesptral coefficient computed over a longer period of time as feature instead of using them individually [81, 140]. In this work we have represented each 30 second block by a single feature vector, of length $2D$, containing the mean and standard deviation of each feature dimension computed from the frames that were selected by the SAD algorithm.

3.6.2 Clean conditions

The obtained classification results for GMM and SVM on the clean dataset are given in Tables 3.3 and 3.4 respectively. The corresponding highest classification accuracies are $76.5 \pm 4.8\%$ and $78.2 \pm 4.1\%$ and are statistically equivalent according to the paired student t-test, i.e. a p-value of 0.36 which is sufficiently larger than the 0.05 significance level [12]. Reducing the sampling frequency from 32 kHz to 16 kHz lowers the computational cost in extracting the features by a factor 2 without changing the overall classification performance. More specifically, the paired student t-test results in a p-value of 0.12 for GMM and 0.87 for SVM, which are both well above the 0.05 significance level, when comparing the best scores of 16 kHz with those of 32 kHz. Further reducing the sampling frequency from 16 kHz to 8 kHz yields statistically significant lower classification accuracies, i.e. p-values of 0.007 and 0.006 for GMM and SVM respectively. In this perspective a sampling frequency of 16 kHz is preferred over the other two settings.

From a computational point of view, the SVM approach is of more interest compared to GMM because of the following two reasons. First, the required bandwidth of the wireless communication between the local nodes and the central processing unit is significantly lower in the SVM topology since only one feature vector must be transmitted every 30 seconds. Secondly, the computational complexity of the SVM approach is also lower as a result that only one feature vector must be classified every 30 seconds. In this setup, the number of training vectors N is 247 resulting into a fixed number of $62,224$ multiplications every 30 seconds for a sampling frequency of 16 kHz, 15 MEL-filters and 14 cepstral coefficients. The required number of multiplications in the GMM topology depends on the number of SAD detected frames in each 30 second block and varies between $19,600$ when only a single frame is SAD detected and $58,800,000$ when all frames are SAD detected. However, the average number of SAD detections per block is rather high making the computational complexity of GMM higher than SVM on an average basis. As a result, the SVM topology with a sampling frequency of 16 kHz is preferred over the other setups and will be examined further in noisy conditions.

Table 3.3: Clean GMM classification results for the different feature parameter settings and the p-value of the paired student t-test for the best classification score of 8 kHz and 16 kHz against the highest classification score of 32 kHz.

# MEL-filters	# cep. coeff.	Sampling frequency 8 kHz	16 kHz	32 kHz
10	7	69.6±3.3%	73.3±4.4%	73.6±5.2%
15	7	70.4±4.2%	73.4±4.8%	74.2±5.3%
15	14	72.8±4.8% (0.0065)	75.1±4.5%	76.5±4.8% (best score)
20	7	70.2±3.1%	72.8±4.9%	74.2±5.3%
20	14	72.7±4.4%	75.5±5.1% (0.1202)	73.0±4.7%

Table 3.4: Clean SVM classification results for the different feature parameter settings and the p-value of the paired student t-test for the best classification score of 8 kHz and 16 kHz against the highest classification score of 32 kHz.

# MEL-filters	# cep. coeff.	Sampling frequency 8 kHz	16 kHz	32 kHz
10	7	68.0±5.5%	72.9±1.7%	71.4±2.8%
15	7	69.3±5.9%	72.8±4.0%	73.5±2.0%
15	14	72.8±5.1% (0.0062)	78.0±2.8% (0.8647)	76.9±2.8%
20	7	70.2±7.4%	72.7±0.7%	71.3±2.4%
20	14	69.3±2.7%	75.3±4.3%	78.2±4.1% (best score)

By examining the confusion matrix of the SVM results for a sampling frequency 16 kHz, 15 MEL-filters and 14 cepstral coefficients given in Table 3.5 it can be noticed that most of the confusion occurs as a result of mutal acoustic information within the different ADLs. A more detailed explanation of the misclassification is as follows:

1. Most confusion occurs within the eating related ADLs (ID 5-8) due to the high degree of shared sound events within these four activities, e.g. utilising cutlery or dishes for instance, making them harder to distinguish from each other.

2. The ADLs related to eating, especially *'setting table'* (ID 7) and *'washing dishes'* (ID 8), together with *'washing hands'* (ID 4) and *'sleeping'* (ID 10) are occasional misclassified as *'dressing'* (ID 9). The latter is a result of the fact that *'dressing'* contains only typically human related sounds, e.g. footsteps, which can be linked to the other activities in the dataset as well.

3. Activities using running tap water on a regular basis are also occasionally confused among each other. These activities are *'brushing teeth'* (ID 1), *'showering'* (ID 2), *'washing hands'* (ID 4) and *'washing dishes'* (ID 8).

Table 3.5: Confusion matrix (in %) of the SVM approach for a sampling frequency of 16 kHz, 15 MEL-filters and 14 cepstral coefficients. The obtained classification performance is $78.0 \pm 2.8\%$.

		Classified label									
	Class ID	1	2	3	4	5	6	7	8	9	10
Ground truth	1	**97.9**	-	-	-	-	-	-	2.1	-	-
	2	3.1	**96.9**	-	-	-	-	-	-	-	-
	3	-	-	**100**	-	-	-	-	-	-	-
	4	4.2	-	-	**91.7**	-	-	-	-	4.2	-
	5	-	1.5	-	-	**77.2**	4.9	4.4	8.3	2.9	1.0
	6	-	0.7	0.7	-	6.3	**55.6**	9.2	19	3.5	4.9
	7	-	-	-	-	4.1	6.6	**73.8**	6.6	9.0	-
	8	1.7	-	-	0.9	16.4	8.6	6.9	**58.6**	6.9	-
	9	-	-	-	-	3.6	-	2.2	0.7	**93.5**	-
	10	-	-	-	-	12.5	5.0	-	-	10.0	**72.5**

Note: See Table 3.1 for the class names.

A possible solution to reduce the confusion within activities containing similar acoustic data is to include spatial features in the classification process as well since some ADLs are typically performed on a specific location in the home environment. For instance, sounds of running tap water recorded in the bathroom are more likely linked to personal hygiene than eating related ADLs.

A simple experiment performed in this context is by including the node number recording with the highest SNR as feature. The latter can be seen as a rudimentary sound source location feature and might improve the classification performance of the WASN. The corresponding confusion matrix is given in Table 3.6 and shows that the classification performance is increased by 3.1% to an overall score of $81.1 \pm 2.3\%$ which is a statistically significant improvement according to the paired student t-test, i.e. p-value of 0.001 and a significance level of 0.05. As one can see, the confusion between ADLs recorded at different locations in the home environment is significantly reduced.

Table 3.6: Confusion matrix (in %) of the SVM approach when the best node number is included as feature for a sampling frequency of 16 kHz, 15 MEL-filters and 14 cepstral coefficients. The obtained classification performance is $81.1 \pm 2.3\%$.

		Classified label									
	Class ID	1	2	3	4	5	6	7	8	9	10
Ground truth	1	**100**	-	-	-	-	-	-	-	-	-
	2	3.1	**96.9**	-	-	-	-	-	-	-	-
	3	-	-	**100**	-	-	-	-	-	-	-
	4	12.5	-	-	**87.5**	-	-	-	-	-	-
	5	-	-	-	-	**81.6**	5.3	5.8	5.3	1.9	-
	6	-	0.7	-	-	7.7	**57.7**	10.6	16.2	2.1	4.9
	7	-	-	-	-	4.9	6.6	**74.6**	8.2	5.7	-
	8	1.7	-	-	-	12.1	9.5	10.3	**66.4**	-	-
	9	-	-	-	-	-	-	-	-	**100**	-
	10	-	-	-	-	10.0	2.5	-	-	15.0	**72.5**

Note: See Table 3.1 for the class names.

3.6.3 Noisy conditions

Figure 3.3 and 3.4 are the results obtained when the background noise source was inserted in the living environment at position A and B respectively. During these experiments, the loudness of the background noise was set at different levels such that the average SNR, here defined as the ratio between the energy of the SAD relevant frames and the CHiME noise data, over all nodes in the combined living room and kitchen varies between 30 dB and 0 dB. The used sampling frequency is 16 kHz and the examined feature parameters are 10 MEL-filters with 7 cepstral coefficients and 20 MEL-filters with 14 cepstral coefficients, i.e. the lowest and highest computational setting respectively. The clean SVM models are used to evaluate the noisy sensor data. Furthermore, it is worth mentioning that during these experiments the same SAD indices are used as in the clean data. The latter is done to eliminate the influence of incorrect sound activity detection on the classification performance of the WASN.

The obtained results indicate that the SNR based dynamic node selection in the combined kitchen and living environment yield only higher classification accuracies compared to choosing a fixed node for the high and medium SNRs, i.e. the range between 30 dB and 10 dB. These SNRs correspond to low noise levels and thus only affecting the node situated closely to the noise source location. The average increase in classification performance for these SNRs compared to the best single microphone solution varies between 0.8% and 2.4% when

the noise source was set a position A and between 3.5% and 4.0% at position B for the parameter settings 20 MEL-filters with 14 cepstral coefficients and 10 MEL-filters with 7 cepstral coefficients respectively. Despite the increased classification accuracy for the dynamic node selection in the range between 30 dB and 10 dB, it is not an statistically significant improvement according to the pairwise student t-test. Further reducing the SNR starts affecting the other nodes in the combined kitchen and home environment as well. This implies that selecting the best node no longer achieves better classification scores. On the other hand, the results also indicate that in noise free conditions one node is typically sufficient for ADL classification. The latter is a result of the relatively small combined kitchen and living environment used in this experimental setup which can be easily covered by one microphone node.

By examining the single node results in more detail it can be seen that node 4 typically achieves the lowest classification scores. This is the node furthest away from the kitchen area where most activities were performed. Furthermore, it can also be observed that node 1 achieves lower classification scores when the background noise source was set relatively close by, i.e. position A, compared to the situation when the noise source is situated further away, i.e. position B. The same trend can be observed for node 2, i.e. lower classification scores when the noise source position was set at position B compared to position A.

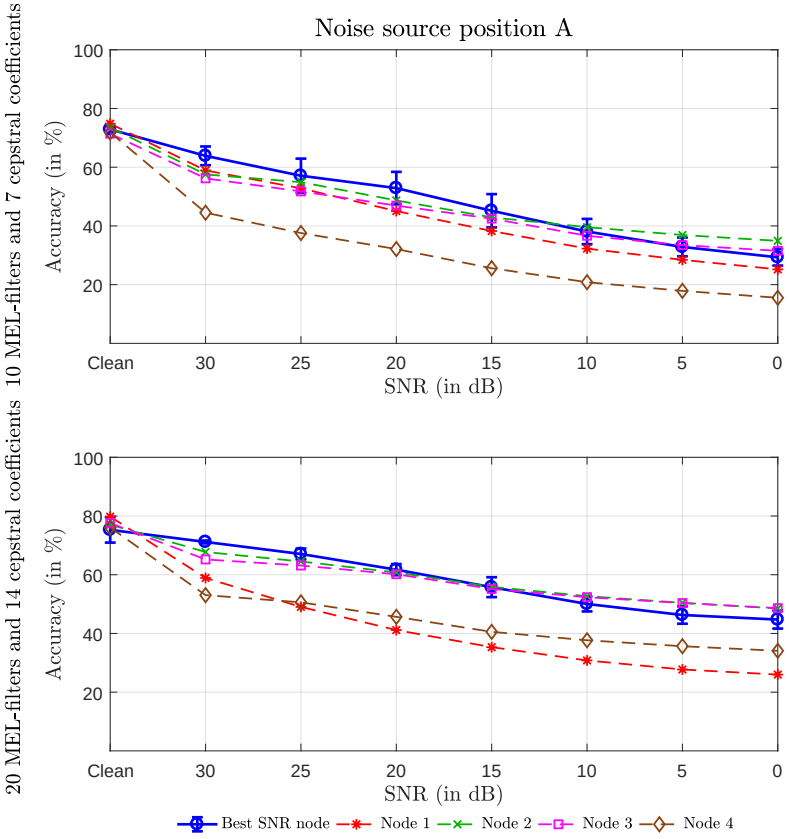

Figure 3.3: Noisy SVM classification results with noise source at position A for the feature parameter setting of 20 MEL-filters with 14 cepstral coefficients (bottom) and 10 MEL-filters and 7 cepstral coefficients (top) respectively. The standard deviation of the best SNR node is also visualised in the plots.

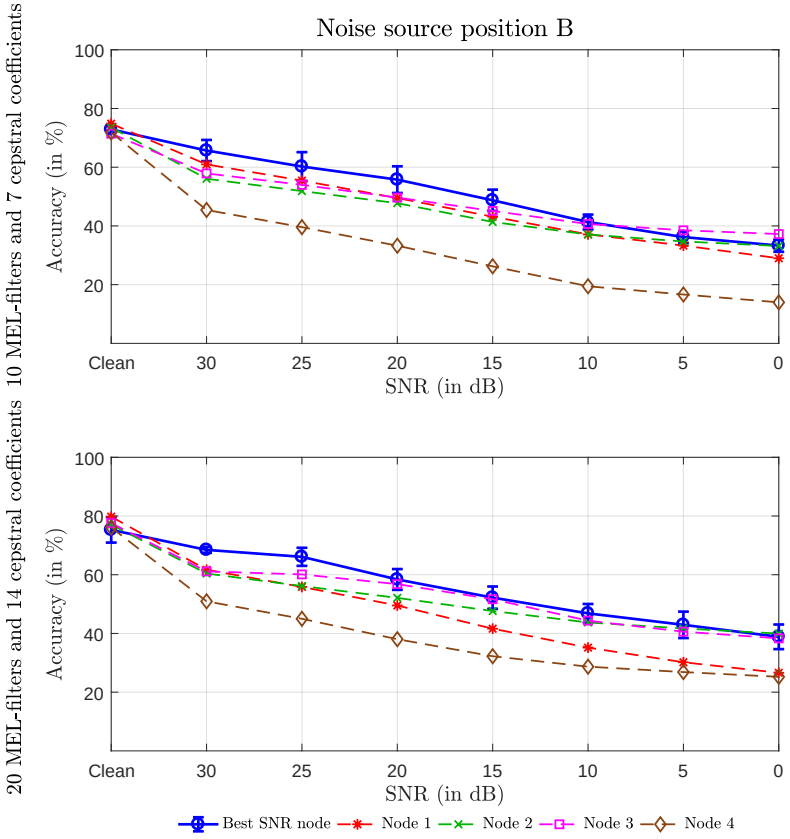

Figure 3.4: Noisy SVM classification results with noise source at position B for the feature parameter setting of 20 MEL-filters with 14 cepstral coefficients (bottom) and 10 MEL-filters and 7 cepstral coefficients (top) respectively. The standard deviation of the best SNR node is also visualised in the plots.

3.7 Conclusions

In this chapter we have examined a WASN for the purpose of classification of activities of daily living w.r.t. computational complexity and noise robustness. The influence of the different MFCC feature extraction configuration settings was examined and evaluated against two baseline classifiers, i.e. GMM and SVM. The highest classification score in clean conditions was obtained for

the SVM classifier for a sampling frequency of 32 kHz, 20 MEL-filters and 14 cepstral coefficients. This parameter setting results in a classification accuracy of $78.2 \pm 4.1\%$. However, by halving the sampling frequency to 16 kHz and reducing the number of MEL-filters to 15 results only in a slight accuracy decrease of 0.2% while the required computational complexity is lowered with a factor of 2.43. Furthermore, the SVM approach in this work has the advantage that the amount of data transfer between the local nodes and the central processing unit is significantly reduced since only one feature vector must be transmitted every 30 seconds. Including the node number as rudimentary localisation feature improves the classification accuracy by 3.1% to $81.1 \pm 2.3\%$.

In addition, the experiments performed under noisy conditions indicate that a WASN yields better classification results for medium and high SNRs compared to single microphone solutions. The average increase in classification accuracies ranges between 0.8% and 2.4% and between 3.5% and 4.0% when the noise source was set at positions A and B respectively. On the other hand, the WASN has no significant advantage in noise free conditions as a result of the relative small combined living environment and kitchen making one microphone node sufficient in this room to collect ADL data.

The main shortcoming in this system architecture is the limited ability to deal with similar acoustic information within the different ADLs. A possible strategy to tackle this issue is by learning the typical isolated sound events occurring in the acoustic data, e.g. *'opening/closing door'*, *'opening/closing drawer'*, *'alarm microwave'*, *'running tap water'*, which can be used for ADL classification. For instance, running tap water in combination with an alarm of a microwave and the sound of cutlery is more likely linked to cooking than doing the dishes. The latter requires a learning strategy capable of dealing with overlapping acoustic events since in time since real-life acoustic data is typically a sequence of individual overlapping sound events. Furthermore, the need for an SAD to determine the relevant parts in the acoustic data for ADL classification is rather impractical due to wide range of interesting sounds and the need of a proper tuning of the parameters. As a result, in the next chapter of this thesis we will focus on learning and modelling (overlapping) sound events occurring in a larger stream of acoustic data without the need of a sound activity detector.

Another shortcoming is the aspect of supervised learning and that the current system architecture is hardly scalable from one location to another, i.e. the trained model for house A cannot be used at house B and vice versa. This implies that for each new location, a set of labelled examples must be collected locally, which is typically a time consuming and labour intensive task, to learn the model specific parameters. As a result, the system architecture in its current setting is difficult to use in real-life settings.

Chapter 4

Weakly supervised multi-label convolutive event modelling and classification

This chapter discusses a weakly supervised and multi-label convolutive learning algorithm based on non-negative matrix deconvolution (NMD). The labelling information is included into the optimisation problem of NMD on a low-resolution and multi-label (LRM) level which reduces the annotation workload on labelling the data significantly. We will show that LRM-NMD is capable of modelling acoustic events from audio labelled on a low-resolution and multi-label level and that it outperforms the investigated baselines. Furthermore, the robustness of LRM-NMD to background noise and overlapping events is also examined during the experiments of this chapter.

The work presented in this chapter was published in:

- Vuegen, L., Karsmakers, P., Vanrumste, B., and Van hamme, H. "Acoustic event classification using low-resolution multi-label non-negative matrix deconvolution." In Journal Audio Engineering Society, AES vol. 66, nr. 5, May 2018, pp. 369-384.
  (**Awarded as the best JAES paper of 2018**)
- Vuegen, L., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Weakly supervised classification of domestic acoustic events for indoor monitoring applications." In proceedings of IEEE Conference on Biomedical and Health Informatics 2018, BHI 2018, Las Vegas, USA, March 2018.

- Meire, M., Vuegen, L., and Karsmakers, P. “The impact of missing labels and overlapping sound events on multi-label multi-instance learning for sound event classification.” Accepted for the proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events, DCASE 2019, New York, United-States, Oct. 2019.

4.1 Introduction

Acoustic event detection and classification (AED/C) have recently become key challenges in the field of acoustic signal processing and are already widely examined for numerous different applications [178]. Over the past few years it has been investigated for surveillance and healthcare related applications [169, 139, 146], tracking and classifying sound sources of military interest [51], automated wildlife observations [54] and diagnoses of industrial machinery [145]. The promise and success of all these applications relies on robust sensing of the environment. Nowadays, collecting acoustic data has become more accessible than ever before due to the ever increasing number of connected devices in our daily lives containing a built-in microphone (e.g. smartphones, smartwatches, tablets, laptops, televisions, remote cameras, etc.). As a result, recordings can be made continuously all day round with minimal effort. However, the development of acoustic event detection and classification algorithms itself typically requires labelled data to learn the model-specific parameters. Accurately labelling the data takes at least the duration of the recording and is thus often the main cost in the process of developing a robust sound event classifier.

The amount of effort that can be spent on labelling the data is expressed by the '*labelling budget*' and defines the maximum number of labels that can be assigned. In situations where the labelling budget is low, or in cases of very large datasets, there are multiple established strategies available in the literature to utilise the abundant amount of unlabelled data with minimal labelling effort. The two best known strategies are '*semi-supervised learning*' and '*active learning*' and will be briefly discussed below.

The majority of the semi-supervised learning (SSL) algorithms are based on the so-called self-training approach which permits to automatically annotate unlabelled data by using a pre-existing model trained on a smaller set of labelled data. The classified instances together with the predicted class labels are added to the training data and the model is retrained. This procedure is then repeated iteratively until a certain target performance is achieved or until no more unlabelled data is available. This allows to deal with unlabelled data and is already examined for various acoustic pattern recognition tasks such as speaker identification [116] and musical instrument recognition [38]. Though self-training is already widely examined, more advanced semi-supervised learning strategies are also available in the literature [26].

Active learning (AL) approaches on the other hand asks for labelling input on data selected by the algorithm from the set of unlabelled data. The annotated labels are then used to train an initial classifier and the remaining unlabelled samples are classified using the learned classifier. A batch of samples with the

lowest classification certainties are presented by the active learning algorithm to the user for verification and are used to update the learned classifier. This procedure is repeated until a stopping criterion is met. Certainty-based sampling has already been widely studied in the field of acoustic pattern recognition [67, 137]. A slightly different approach of active learning is when the classified instances are directly observed by the user and corrected if necessary, e.g. lights won't turn on in a voice steered home automation system after the command *'turn on the lights'* is spoken. This strategy does not require a certainty based metric for sampling and is recently successfully applied for acoustic pattern recognition tasks such as recognising speech commands from dysarthric speech [57].

All the above-mentioned techniques rely on a classifier either for sampling or prediction of the unlabelled data. We aim at learning the model-specific parameters from both the labelled and unlabelled instances directly by utilising the internal structure of the data. An already widely examined technique for structure discovery in the field of acoustic signal processing are the so-called '*non-negative matrix factorisation*' (NMF) based approaches and their extensions. The main objective of NMF is to decompose an all positive data matrix into a product of two (or more) non-negative lower rank submatrices called factors. As a result of the restriction to positive values only, the factors tend to model the data matrix as a linear combination of additive components and thereby reveal the underlying latent structure. In the experiments of this chapter, we will focus on using a convolutive variant of NMF, called '*non-negative matrix deconvolution*' (NMD), which is capable of identifying components with a temporal structure as well [149]. Although both NMF and NMD tend to a parts-based representation of the data, there is no absolute guarantee for this behaviour. In this work we present an extension to NMD where the parts-based solutions are promoted by incorporating the available class labels in the learning procedure of NMD. Label information must only be available at a '*low-resolution*', i.e. an indication of which events have occurred in an segment, without identifying beginnings nor endings of the individual events. This also leads to a '*multi-label*' approach in training and classification: a sound segment can contain multiple (different) events which may or may not overlap with each other in time. Lastly, we will assume that not all occurring events in an segment are annotated as occurring. In other words, we assume annotations are incomplete, i.e. '*weakly supervision*', and available at a low temporal resolution, which significantly reduces the annotation cost.

The remainder of this chapter is organised as followed: in Section 4.2 convolutive event modelling is thoroughly explained together with the adopted changes to include the low-resolution multi-labelling information into the learning procedure. The experimental specific details are given in Section 4.3 while the experiments

together with the obtained classification results are discussed in Section 4.4. Finally, the discussion and concluding remarks of this research are given in Section 4.5 and 4.6 respectively.

4.2 Convolutive event modelling

4.2.1 Introduction to non-negative matrix deconvolution

Non-negative matrix deconvolution (NMD), also known as convolutive non-negative matrix factorisation (CNMF) or convolutive sparse coding (CSC), is an extension of non-negative matrix factorisation (NMF) and is capable of identifying components with a temporal structure [149, 101]. The main objective of NMD is to decompose an all-positive observation matrix $\mathbf{Y} \in \mathbb{R}_{\geq 0}^{B \times F}$, e.g. a B-dimensional magnitude spectrogram of duration F in case of acoustic processing, into the convolution between a set of temporal basis matrices $\mathbf{A}_t \in \mathbb{R}_{\geq 0}^{B \times L}$, with $t \in [1, T]$, and its activation matrix $\mathbf{X} \in \mathbb{R}_{\geq 0}^{L \times F}$, and is formulated by

$$\mathbf{Y} \approx \mathbf{\Psi} = \sum_{t=1}^{T} \mathbf{A}_t \overrightarrow{\mathbf{X}}^{(t-1)} . \tag{4.1}$$

The operator $\overrightarrow{(\cdot)}^{t}$ denotes a matrix shift of t entries to the right. Columns that are shifted out at the right are discarded, while zeros are shifted in from the left. Conversely, the operation $\overleftarrow{(\cdot)}^{t}$ shifts columns to the left, with zero filling on the right and removal of the shifted out columns on the left. It is now clear how to read Equation 4.1 and it can be noticed that NMD turns into regular NMF for the special case $T = 1$.

The complete set of basis data is described by combining all temporal basis matrices $\mathbf{A}_t$ into a global three-way tensor $\mathbf{A} \in \mathbb{R}_{\geq 0}^{B \times L \times T}$. Each l-th slice of $\mathbf{A}$, denoted further as $\mathbf{A}_{(l)}$, then contains the temporal basis data of the lth-component over time t and can be interpreted as one of the additive time-frequency dictionary elements describing the underlying structure in $\mathbf{Y}$. In other words, the reconstructed data $\mathbf{\Psi}$ is obtained by summing the L convolutions obtained between $\mathbf{A}_{(l)}$ and the l-th row in $\mathbf{X}$. Hence, each row in $\mathbf{X}$ can thus be seen as an activation vector for the corresponding time-frequency dictionary element $\mathbf{A}_{(l)}$.

The objective of the decomposition is to estimate $\mathbf{A}$ and $\mathbf{X}$ such that the error between $\mathbf{Y}$ and $\mathbf{\Psi}$ is minimised. Different error measures are proposed in the

literature [102] and we have chosen to use the generalised Kullback-Leibler divergence (beta divergence for $\beta = 1$) which is given by

$$D_{KL}(\mathbf{Y}\|\mathbf{\Psi}) = \sum_{(b,f)\in(B,F)} (\mathbf{Y}_{bf} \log \frac{\mathbf{Y}_{bf}}{\mathbf{\Psi}_{bf}} - \mathbf{Y}_{bf} + \mathbf{\Psi}_{bf}). \tag{4.2}$$

Although NMF and NMD favour a sparse and parts-based representation of the observation data $\mathbf{Y}$ there is no absolute guarantee for this behaviour [71]. Therefore, an additional weighted L_1-norm penalty is typically applied to the activations to control the sparseness. Hence, the total cost function is expressed as

$$\min_{\mathbf{A},\mathbf{X}} \Big(D_{KL}(\mathbf{Y}||\mathbf{\Psi}) + \lambda \|\mathbf{X}\|_1 \Big), \tag{4.3}$$

with $\lambda\|\mathbf{X}\|_1$ denoting the weighted sparsity penalty term. Increasing λ will tend to sparser solutions. It is worth to note that this approach of sparsity penalisation is not scale-invariant and can thus be trivially minimised by scaling $\mathbf{A}$ up and $\mathbf{X}$ down without altering $D_{KL}(\mathbf{Y}||\mathbf{\Psi})$. A solution to overcome this scaling misbehaviour is to minimise Equation 4.3 under the constraint that $\mathbf{A}_{(l)}$, $\forall l \in L$, is normalised to unit Frobenius norm as proposed in [98, 125]. Different strategies are available in the literature to minimise Equation 4.3, however the most commonly used approach in the domain of NMF and NMD are the so-called multiplicative updates, and are

$$\mathbf{X} \leftarrow \mathbf{X} \circ \frac{\sum_{t=1}^{T} \mathbf{A}_t^\top \overleftarrow{\left(\frac{\mathbf{Y}}{\mathbf{\Psi}}\right)}^{(t-1)}}{\sum_{t=1}^{T} \mathbf{A}_t^\top \overleftarrow{\mathbf{1}_u}^{(t-1)} + \lambda}, \tag{4.4}$$

for the activations and

$$\mathbf{A}_t \leftarrow \mathbf{A}_t \circ \frac{\left(\frac{\mathbf{Y}}{\mathbf{\Psi}}\right) \overrightarrow{\mathbf{X}}^{(t-1)^\top} + \mathbf{A}_t \circ \left(\mathbf{1}_v \left[\mathbf{A}_t \circ \left(\mathbf{1}_u \overrightarrow{\mathbf{X}}^{(t-1)^\top} \right) \right] \right)}{\mathbf{1}_u \overrightarrow{\mathbf{X}}^{(t-1)^\top} + \mathbf{A}_t \circ \left(\mathbf{1}_v \left[\mathbf{A}_t \circ \left[\left(\frac{\mathbf{Y}}{\mathbf{\Psi}}\right) \overrightarrow{\mathbf{X}}^{(t-1)^\top} \right] \right] \right)}, \forall t \in [1, T], \tag{4.5}$$

for the temporal basis where $\circ$ denotes the Hadamard product (an elementwise multiplication) and both $\mathbf{1}_u$ and $\mathbf{1}_v$ are all-one matrices of dimensions $B \times F$ and $B \times B$ respectively. Each lth-component over time t in the temporal basis data is then normalised to unit Frobenius norm after each update by using $\mathbf{A} = \left[\frac{\mathbf{A}_{(1)}}{\|\mathbf{A}_{(1)}\|_F}, \frac{\mathbf{A}_{(2)}}{\|\mathbf{A}_{(2)}\|_F}, \ldots, \frac{\mathbf{A}_{(L)}}{\|\mathbf{A}_{(L)}\|_F} \right]$ to prevent scaling misbehaviour.

4.2.2 Low-resolution multi-label NMD

In this chapter we introduce an extended version of NMD, called low-resolution multi-label non-negative matrix deconvolution (LRM-NMD) where both the observation data and the available labelling information are used during training. The labelling information is included into the learning procedure by partitioning the observation data $\mathbf{Y}^{[o]}$ into larger segments, i.e. $\mathbf{Y}^{[o]} = [\mathbf{Y}_1^{[o]}, \mathbf{Y}_2^{[o]}, \ldots, \mathbf{Y}_J^{[o]}]$, all containing a sequence of multiple and potentially overlapping sound events from different classes. The latter is done by assigning a semantic label vector $\mathbf{y}_j^{[s]} \in \{0,1\}^C$ to each $\mathbf{Y}_j^{[o]}$ indicating the sound classes, or even a subset of the sound classes, that take place in the corresponding segment as shown in Figure 4.1.

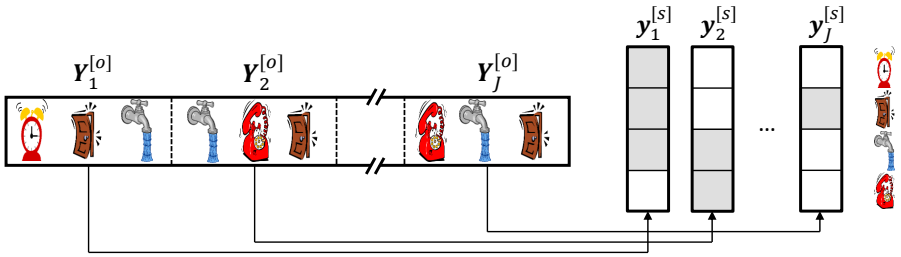

Figure 4.1: Graphical visualisation of the low-resolution multi-labelling strategy (LRM). Each segment $\mathbf{Y}_j^{[o]}$ is labelled by its low-resolution multi-label vector $\mathbf{y}_j^{[s]}$ indicating the sound classes, or a subset of sound classes, that take place in the corresponding segment. As one can see, not all events in the segments must be labelled as occurring.

This approach of labelling reduces the amount of workload in annotating the data significantly. The time duration of the segments may vary, i.e. they don't have to be of equal length, and defines the resolution of the labelling. More specifically, longer segments corresponds to a lower resolution compared to shorter segments. Formally, the low-resolution multi-label vectors are defined by

$$\mathbf{y}_{(c,j)}^{[s]} = \begin{cases} 1 & \text{if segment } j \text{ contains data from sound class } c, \\ 0 & \text{otherwise,} \end{cases} \tag{4.6}$$

where $\mathbf{y}_{(c,j)}^{[s]}$ denotes the cth-element in $\mathbf{y}_j^{[s]}$. The main objective of LRM-NMD is still to decompose the observations $\mathbf{Y}_j^{[o]}, \forall j \in J$, by using Equation 4.1 but

with respect to

$$\mathbf{y}_j^{[s]} \approx \boldsymbol{\psi}_j^{[s]} = \mathbf{A}^{[s]}\mathbf{X}_j\mathbf{1}_j, \tag{4.7}$$

with $\mathbf{1}_j$ an all-one column vector of length F_j and $\mathbf{A}^{[s]}$ acting as a labelling matrix for the temporal basis data $\mathbf{A}^{[o]}$ and is defined by

$$\mathbf{A}^{[s]}_{(c,l)} = \begin{cases} 1 & \text{if } \mathbf{A}^{[o]}_{(l)} \text{ belongs to sound class } c, \\ 0 & \text{otherwise.} \end{cases} \tag{4.8}$$

This additional factorisation favours decompositions such that there is a resemblance with the labelling information as well. More specifically, LRM-NMD encourages that the events in segment $\mathbf{Y}_j^{[o]}$, labelled by $\mathbf{y}_j^{[s]}$, are decomposed with their subset of temporal basis data given the labelling matrix $\mathbf{A}^{[s]}$ and thereby further improving the parts-based representation in $\mathbf{A}^{[o]}$. A graphical example of LRM-NMD is given in Figure 4.2 where events of three different sound classes in $\mathbf{Y}^{[o]}$ are decomposed into three temporal basis matrices, each four frames long, and their activations.

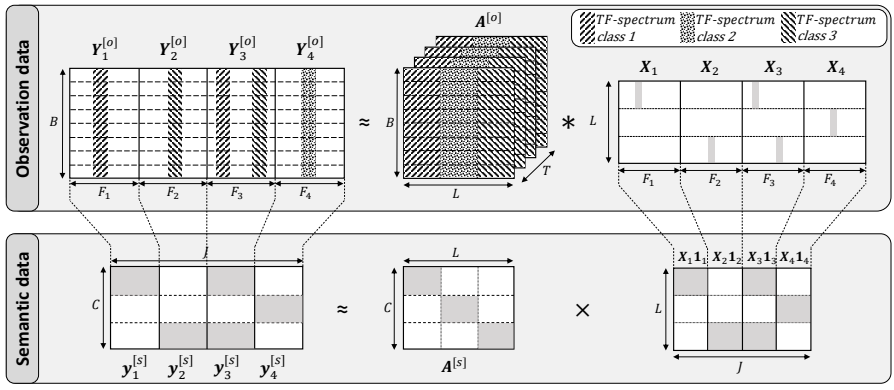

Figure 4.2: A graphical example of LRM-NMD where events of three different sound classes ($C = 3$) in $\mathbf{Y}^{[o]}$ are decomposed into a set of temporal basis matrices ($L = 3$) each four frames long ($T = 4$) and their activations. The operators $*$ and $\times$ denote the convolutive factorisation according to Eq. 4.1 and a standard matrix multiplication respectively. The upper part visualises the convolutive modelling of events w.r.t. the low-resolution multi-label vectors given in the lower part. The learned temporal basis data $\mathbf{A}^{[o]}$ can be interpreted as a set of dictionary elements describing the underlying structure in the observation data $\mathbf{Y}^{[o]}$ while $\mathbf{A}^{[s]}$ acts as a labelling matrix of $\mathbf{A}^{[o]}$ indicating to which sound class each learned dictionary element corresponds.

The cost function of LRM-NMD is expressed by

$$\min_{\mathbf{A}^{[o]},\mathbf{A}^{[s]},\mathbf{X}} \left[\sum_{j=1}^{J} \Big(D_{KL}(\mathbf{Y}_j^{[o]} \| \mathbf{\Psi}_j^{[o]}) + \lambda \|\mathbf{X}_j\|_1 + \eta D_{KL}(\mathbf{y}_j^{[s]} \| \boldsymbol{\psi}_j^{[s]}) \Big) \right], \tag{4.9}$$

with η as a semantic balancing parameter between the acoustic observation data and the low-resolution mult-label information. Larger values of η yield more focus on the labelling information but with risk of overfitting. Conversely, LRM-NMD reduces to conventional NMD when $\eta = 0$. Like for NMD, iterative update formulas for $\mathbf{X}_j$, $\forall j \in J$, $\mathbf{A}^{[o]}$ and $\mathbf{A}^{[s]}$ can be derived to minimise the cost function given by Equation 4.9 using the same procedure as proposed in [101]. The corresponding updates are

$$\mathbf{X}_j \leftarrow \mathbf{X}_j \circ \frac{\left(\sum_{t=1}^{T} \mathbf{A}_t^{[o]\top} \overleftarrow{\left(\frac{\mathbf{Y}_j^{[o]}}{\mathbf{\Psi}_j^{[o]}} \right)^{(t-1)}} \right) + \eta \left(\mathbf{A}^{[s]\top} \left(\frac{\mathbf{y}_j^{[s]}}{\boldsymbol{\psi}_j^{[s]}} \right) \mathbf{1}_j^\top \right)}{\left(\sum_{t=1}^{T} \mathbf{A}_t^{[o]\top} \overleftarrow{\mathbf{1}_{u_j}^{(t-1)}} + \lambda \right) + \eta \left(\mathbf{A}^{[s]\top} (\mathbf{1}_w) \mathbf{1}_j^\top \right)}, \tag{4.10}$$

$$\mathbf{A}_t^{[o]} \leftarrow \mathbf{A}_t^{[o]} \circ \frac{\sum_{j=1}^{J} \left(\left(\frac{\mathbf{Y}_j^{[o]}}{\mathbf{\Psi}_j^{[o]}} \right) \overrightarrow{\mathbf{X}_j^{(t-1)}}^\top + \mathbf{A}_t \circ \left(\mathbf{1}_{v_j} \left[\mathbf{A}_t \circ \left(\mathbf{1}_{u_j} \overrightarrow{\mathbf{X}^{(t-1)}}^\top \right) \right] \right) \right)}{\sum_{j=1}^{J} \left(\mathbf{1}_{u_j} \overrightarrow{\mathbf{X}_j^{(t-1)}}^\top + \mathbf{A}_t \circ \left(\mathbf{1}_{v_j} \left[\mathbf{A}_t \circ \left[\left(\frac{\mathbf{Y}}{\mathbf{\Psi}} \right) \overrightarrow{\mathbf{X}^{(t-1)}}^\top \right] \right] \right) \right)}, \forall t \in [1, T], \tag{4.11}$$

and

$$\mathbf{A}^{[s]} \leftarrow \mathbf{A}^{[s]} \circ \frac{\sum_{j=1}^{J} \left(\left(\frac{\mathbf{y}_j^{[s]}}{\boldsymbol{\psi}_j^{[s]}} \right) \left(\mathbf{X}_j \mathbf{1}_j \right)^\top \right)}{\sum_{j=1}^{J} \left(\mathbf{1}_w \left(\mathbf{X}_j \mathbf{1}_j \right)^\top \right)}, \tag{4.12}$$

for the activations $\mathbf{X}_j$, $\forall j \in J$, the dictionary matrix $\mathbf{A}^{[o]}$ and the labelling matrix $\mathbf{A}^{[s]}$ respectively with $\mathbf{1}_w$ denoting an all-one column vector of length C. Furthermore, it is worth noting that the additional term $D_{KL}(\mathbf{y}_j^{[s]} \| \boldsymbol{\psi}_j^{[s]})$ in the cost function of LRM-NMD does not change the update of $\mathbf{A}^{[o]}$ but that we only have rewritten the numerator and denominator into a sum over the J segments. The pseudo code for the complete training procedure of LRM-NMD is given by Algorithm 1. The derivations of Equations 4.10, 4.11 and 4.12 can be found in Appendix 7.1.1, 7.1.2 and 7.1.3 respectively.

Algorithm 1 LRM-NMD training procedure

Inputs: $\mathbf{Y}_j^{[o]}$, $\mathbf{y}_j^{[s]}$, λ and η with $j \in [1, J]$
Initialise: $\mathbf{A}^{[o]}$, $\mathbf{A}^{[s]}$ and $\mathbf{X}_j$
Normalise: $\mathbf{A}_{(l)}^{[o]} \leftarrow \mathbf{A}_{(l)}^{[o]} / \|\mathbf{A}_{(l)}^{[o]}\|_F$, $\forall l \in L$
repeat
  for $j = 1 : J$ **do**
    $\boldsymbol{\Psi}_j^{[o]} = \sum_{t=1}^{T} \mathbf{A}_t^{[o]} \overrightarrow{\mathbf{X}_j}^{(t-1)}$
    $\mathbf{X}_j \leftarrow \mathbf{X}_j \circ \frac{\left(\sum_{t=1}^{T} \mathbf{A}_t^{[o]\top} \left(\frac{\mathbf{Y}_j^{[o]}}{\boldsymbol{\Psi}_j^{[o]}} \right) \right) + \eta \left(\mathbf{A}^{[s]\top} \left(\frac{\mathbf{y}_j^{[s]}}{\boldsymbol{\psi}_j^{[s]}} \right) \mathbf{1}_j^\top \right)}{\left(\sum_{t=1}^{T} \mathbf{A}_t^{[o]\top} \overleftarrow{\mathbf{1}_{u_j}}^{(t-1)} + \lambda \right) + \eta \left(\mathbf{A}^{[s]\top} (\mathbf{1}_w) \mathbf{1}_j^\top \right)}$
    $\boldsymbol{\Psi}_j^{[o]} = \sum_{t=1}^{T} \mathbf{A}_t^{[o]} \overrightarrow{\mathbf{X}_j}^{(t-1)}$
  end for
  $\mathbf{A}_t^{[o]} \leftarrow \mathbf{A}_t^{[o]} \circ \frac{\sum_{j=1}^{J} \left(\left(\frac{\mathbf{Y}_j^{[o]}}{\boldsymbol{\Psi}_j^{[o]}} \right) \overrightarrow{\mathbf{X}_j}^{(t-1)\top} + \mathbf{A}_t \circ \left(\mathbf{1}_{v_j} \left[\mathbf{A}_t \circ \left(\mathbf{1}_{u_j} \overrightarrow{\mathbf{X}_j}^{(t-1)\top} \right) \right] \right) \right)}{\sum_{j=1}^{J} \left(\mathbf{1}_{u_j} \overrightarrow{\mathbf{X}_j}^{(t-1)\top} + \mathbf{A}_t \circ \left(\mathbf{1}_{v_j} \left[\mathbf{A}_t \circ \left[\left(\frac{\mathbf{Y}}{\boldsymbol{\Psi}} \right) \overrightarrow{\mathbf{X}_j}^{(t-1)\top} \right] \right] \right) \right)}, \forall t \in [1, T],$
  $\mathbf{A}_{(l)}^{[o]} \leftarrow \mathbf{A}_{(l)}^{[o]} / \|\mathbf{A}_{(l)}^{[o]}\|_F$, $\forall l \in L$
  for $j = 1 : J$ **do**
    $\boldsymbol{\Psi}_j^{[o]} = \sum_{t=1}^{T} \mathbf{A}_t^{[o]} \overrightarrow{\mathbf{X}_j}^{(t-1)}$
    $\boldsymbol{\psi}_j^{[s]} = \mathbf{A}^{[s]} \mathbf{X}_j \mathbf{1}_j$
  end for
  $\mathbf{A}^{[s]} \leftarrow \mathbf{A}^{[s]} \circ \frac{\sum_{j=1}^{J} \left(\left(\frac{\mathbf{y}_j^{[s]}}{\boldsymbol{\psi}_j^{[s]}} \right) \left(\mathbf{X}_j \mathbf{1}_j \right)^\top \right)}{\sum_{j=1}^{J} \left(\mathbf{1}_w \left(\mathbf{X}_j \mathbf{1}_j \right)^\top \right)}$
  for $j = 1 : J$ **do**
    $\boldsymbol{\psi}_j^{[s]} = \mathbf{A}^{[s]} \mathbf{X}_j \mathbf{1}_j$
  end for
until convergence **or** when max. iterations is reached
return $\mathbf{A}^{[o]}$, $\mathbf{A}^{[s]}$ and $\mathbf{X}_j, \forall j \in J$

4.2.3 Finding activations from the basis data

Once a representative set of basis data is learned from the training data, the task of LRM-NMD becomes decomposing unseen observation data $\mathbf{Y}^{[o]}$ such that the estimated activations $\mathbf{X}$ minimise the cost function given by Equation 4.3 under the fixed learned basis data $\mathbf{A}^{[o]}$. Like in training, the activations are obtained by applying iterative updates to $\mathbf{X}$ while keeping $\mathbf{A}^{[o]}$ fixed. It is worth mentioning that the updates are done by Equation 4.4, instead of Equation 4.10, since the class labels are hidden and thereby cannot be used.

The obtained activations are very well suited for the purpose of structure discovery in $\mathbf{Y}^{[o]}$ due to the parts-based representation of $\mathbf{A}^{[o]}$. Generally, the degree of activation in $\mathbf{X}$ indicates where events in $\mathbf{Y}^{[o]}$ occur while the labelling matrix $\mathbf{A}^{[s]}$ reveals the sound classes of the detected events. The pseudo code of activation estimation is given by Algorithm 2. However, further processing on the activations is required to make a decision over which event has occurred and will discussed in more detail in Section 4.3.4.

Algorithm 2 LRM-NMD evaluation procedure

Inputs: $\mathbf{Y}^{[o]}$, $\mathbf{A}^{[o]}$, $\mathbf{A}^{[s]}$ and λ
Initialise: $\mathbf{X}$
repeat
$$\mathbf{\Psi}^{[o]} = \sum_{t=1}^{T} \mathbf{A}_t^{[o]} \overrightarrow{\mathbf{X}}^{(t-1)}$$
$$\mathbf{X} \leftarrow \mathbf{X} \circ \frac{\sum_{t=1}^{T} \mathbf{A}_t^{[o]\top} \overleftarrow{\left(\frac{\mathbf{Y}^{[o]}}{\mathbf{\Psi}^{[o]}}\right)}^{(t-1)}}{\sum_{t=1}^{T} \mathbf{A}_t^{[o]\top} \overleftarrow{\mathbf{1}_u}^{(t-1)} + \lambda}$$
$$\mathbf{\Psi}^{[o]} = \sum_{t=1}^{T} \mathbf{A}_t^{[o]} \overrightarrow{\mathbf{X}}^{(t-1)}$$
until convergence **or** when max. iterations is reached
return $\mathbf{X}$

4.3 Experimental specific details

4.3.1 From raw data to features

The LRM-NMD method will be evaluated in a context of monitoring persons at home based on acoustics. For this purpose the publicly available NAR dataset is selected. This dataset contains a set of real-life isolated domestic audio events, collected with a humanoid robot Nao, and is recorded especially for acoustic classification benchmarking in domestic environments [76, 77].

In total 42 different sound classes were recorded and can be categorised into *'kitchen related events'*, *'office related events'*, *'non-verbal events'* and *'verbal events'*. The verbal events are not used in this research since we are not interested in detecting and classifying speech commands such as *'left'*, *'right'*, *'start'*, *'stop'*, etc. Therefore, the dataset is reduced to 20 sound classes each containing 20 or 21 recordings of an isolated event. A detailed overview of the used data together with the average duration per sound class is given in Table 4.1.

Furthermore, to examine the influence of background noise we have artificially mixed the clean events with natural environmental sounds as well, i.e. recordings of *'rain'*, *'hail'* and *'blowing wind'*, which are a realistic type of background noise in case of home monitoring. The examined SNR levels are *'clean'*, 20, 10, 5, 3 and 0 dB.

A good choice of non-negative and approximately additive acoustic features which are typically used in the domain NMF and NMD are the so-called MEL-magnitude spectrograms as discussed in Section 2.2.1 [149, 58]. In this experiment, the MEL-magnitude spectrograms spanning 40 bands are computed using a frame length of 25 ms and a frame shift of 10 m. The used filter bank is constructed such that the begin frequency of the first MEL-filter and the end frequency of the last MEL-filter correspond to the frequency range of the microphone, i.e. 300 Hz and 18 kHz. Furthermore, all events are normalised to unit Frobenius norm over the duration of the recording to prevent the higher energetic events from dominating the training cost function to the disadvantage of those with lower energy during training.

The features for the training and test sets are obtained by randomly sampling events from each sound class in the NAR dataset with a ratio of 60% training data and 40% test data. This implies that each class is represented with twelve or thirteen training examples and eight test examples resulting into class balanced training and test sets. In total four independent folds are generated to examine the statistical relevance of the results as well.

Table 4.1: Overview of the used NAR dataset.

Cat.	ID	Sound class	# Events	Avg. duration in (sec.)
Kitchen	**1**	Alarm microwave	21	0.52 ± 0.06
	2	Alarm refrigerator	21	0.55 ± 0.03
	3	Closing microwave	21	0.37 ± 0.07
	4	Opening microwave	21	0.28 ± 0.05
	5	Drawer	21	1.04 ± 0.11
	6	Taking cutlery	21	0.35 ± 0.06
	7	Toaster	21	0.45 ± 0.10
	8	Running tapwater	21	1.78 ± 0.43
Non-verbal	**9**	Coughing	21	0.84 ± 0.09
	10	Fingersnap	20	0.23 ± 0.06
	11	Handclap	20	0.36 ± 0.08
	12	Tongue click	20	0.18 ± 0.06
Office	**13**	Doorknock	20	0.32 ± 0.06
	14	Locking/unlocking door	20	0.29 ± 0.08
	15	Opening door	20	0.29 ± 0.12
	16	Closing door	20	0.44 ± 0.05
	17	Moving chair	21	0.97 ± 0.10
	18	Tearing paper	20	0.46 ± 0.08
	19	Opening zipper 1	20	0.34 ± 0.04
	20	Opening zipper 2	20	0.40 ± 0.06

Note: The verbal events of the NAR dataset are not listed in this table. See [77] for a complete overview.

4.3.2 Basis initialisation

Like NMF, Algorithm 1 in Section 4.2.2 can only be proven to find a local optimum of its cost function, and not a global one, and is therefore sensitive to the initialisation of its factors [101, 102]. Careful initialisation of $\mathbf{A}^{[o]}$, $\mathbf{A}^{[s]}$ and $\mathbf{X}$ can improve the speed and accuracy of the factorisation, as it can produce faster convergence to a better local minimum. Although random initialisation with small positive numbers is by far the most commonly used method in the

literature, we have chosen an exemplar-based initialisation which has been found to be effective as well in the case of acoustic event modelling [58, 95]. Hence, the temporal basis data $\mathbf{A}^{[o]}$ is initialised with one example per sound class randomly sampled from the training data and the labelling matrix $\mathbf{A}^{[s]}$ is initialised according to Equation 4.8. The silence frames before and after the events are trimmed and the shorter events are padded with small positive values uniformly distributed between the range ϵ (floating-point relative accuracy) and 10^{-4} after the normalisation such that they are all of equal length. One exception is *'running tapwater'*, where the length is limited to the maximum length of the other chosen events, i.e. 40 frames in this work, due to stationarity in its data. The overall dimensions of $\mathbf{A}^{[o]}$ are thus $B = 40$, $L = 20$ and $T = 40$. The activations $\mathbf{X}$ on the other hand are initialised with a constant value $1/L$.

4.3.3 Background modelling

The implementation of a background noise model is done in a straightforward manner by adding isolated noise examples to the basis data $\mathbf{A}^{[o]}$, without linking them to a specific sound class given the labelling matrix $\mathbf{A}^{[s]}$, and keeping them fixed during training. Here, only one example for each type of noise is added to $\mathbf{A}^{[o]}$ resulting into four additional dictionary elements representing the noise data, i.e. *'Nao fan noise'*, *'rain'*, *'hail'* and *'blowing wind'* respectively. As a result of the background modelling, the total number of temporal basis vectors in $\mathbf{A}^{[o]}$ is increased to $L = 24$. In addition, the sparsity penalty weights corresponding to the background noise basis data are set to zero and the noise activations are restricted to be constant over periods of one second to mimic the stationarity and slow time-varying properties of the background noise. This form of parameter tying can be expressed in the optimisation problem given by Equation 4.9 and is tantamount to overwriting the rows in the numerator and denominator of Equation 4.10 corresponding to the noise activations with their mean value computed from non-overlapping one second windows.

4.3.4 Event classification

The classification task is implemented by feeding features of isolated non-overlapping events to the LRM-NMD evaluation procedure given by Algorithm 2. The obtained activations are summed over time and the predicted class label $\hat{C}$ is defined by the temporal basis matrix $\mathbf{A}_{(l)}$ with the highest accumulated activation in which the background noise related activations are disregarded. The used metric to evaluate the classification performance is classification accuracy which defines the percentage of correctly classified events.

4.4 Experiments and results

4.4.1 Experimental setup

The experiments of this chapter will mainly focus on investigating the relation between the number of labelled data and the classification performance of LRM-NMD. This relation is examined in two different operating modes, i.e. '*single-label learning*' and '*low-resolution multi-label learning*', and will be discussed in more detail in Sections 4.4.3 and 4.4.4 respectively. The main difference between both operating modes is that in single-label learning each $\mathbf{Y}_j^{[o]}$, $\forall j \in J$, is an isolated sound event, while in low-resolution multi-label learning each $\mathbf{Y}_j^{[o]}$, $\forall j \in J$, is a segment spanning multiple sound events as shown in Figure 4.3. The latter operating mode is of more interest in case of real-life settings, e.g. home-monitoring, since it reduces the workload on labelling the training data significantly. In addition, the noise robustness of both operating modes is also examined and in case of low-resolution multi-label learning we will examine the influence of overlapping events in the annotated segments as well. Furthermore, the behaviour of the semantic balancing parameter (η) and the sparsity penalty (λ) on the classification performance of LRM-NMD is also investigated.

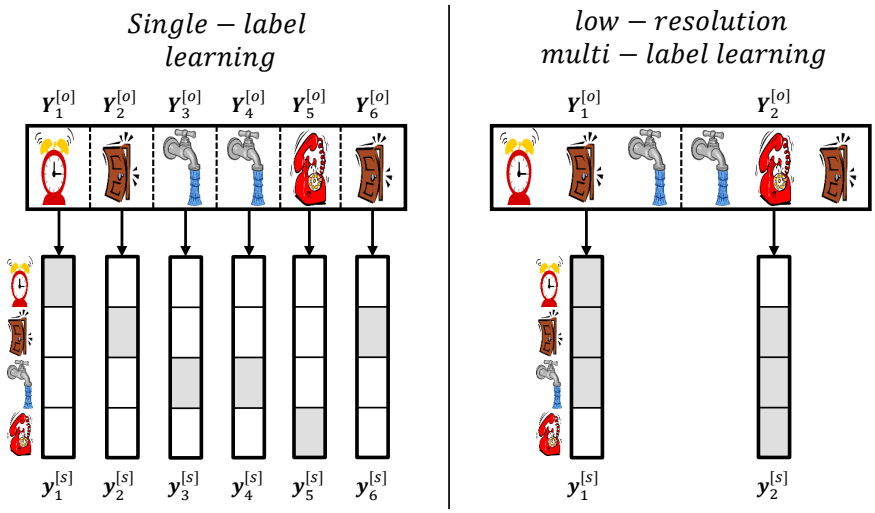

Figure 4.3: Graphical visualisation of the difference between single-label learning mode (left) vs. low-resolution multi-label learning mode (right).

4.4.2 Baselines

This section briefly discusses the used baselines, together with their implementation specific details, which have been used to benchmark LRM-NMD. The used baselines are chosen w.r.t. the methods described in [76, 77] and are:

1. **Gaussian mixture model** (GMM): is a weighted sum of M-multivariate Gaussian distributions modelling the feature space $\mathbf{Y}^{[o]}$ as described in Section 2.3.1. During classification, the objective is to find the class model $\boldsymbol{\theta}_c$ with the highest a-posteriori probability $p(\mathbf{Y}_{te}^{[o]}|\boldsymbol{\theta}_c)$ given the test features $\mathbf{Y}_{te}^{[o]}$. As opposed to NMD, GMMs are typically used in combination with MEL-frequency cepstral coefficients (MFCC) instead of the MEL-magnitude features due to their ability to represent the amplitude spectrogram in a compact and decorrelated form [136]. The MFCCs are computed from the MEL-features by applying a discrete cosine transform (DCT) on the log-transformed MEL-magnitudes as explained in Section 2.2.1.

   Here, 13th order MFCCs are computed, without the deltas (rate of change) and delta-deltas (acceleration), and are normalised such that the complete training set has zero mean and unit standard deviation in each cepstral dimension. The number of mixture components is tuned on a separate development set obtained by further partitioning the test data into two equal sized subsets, alternately serving for development and test respectively. More specifically, during the first run 50% of the test data is used as development set and the remaining 50% as test data. The number of mixtures are tuned on the development set and evaluated against the test set. During the second run the same procedure is performed but with switched roles of the subsets, i.e. first 50% is used for testing and the remaining 50% for development. The global classification accuracy of the fold is computed by averaging the classification scores of the two sub results. This procedure is done to keep a fixed training set since we want to examine the relation between the amount of training examples and the classification performance. Furthermore, all GMMs are modelled with diagonal covariance matrices to prevent overfitting due to the limited amount of training data.

2. **Support vector machine** (SVM): is a binary classifier formally characterised by a separating hyperplane such that the margin, i.e. distance, between the two classes is maximised as explained in Section 2.3.2. Here, 1v1 is used as coding scheme to expand this binary classification problem into a multiclass variant and the overall classification score is computed by a majority vote over the sub results [24].

In contrast to GMM, SVM typically uses the normalised mean and standard deviation of each MFCC dimension as acoustic features instead of using them individually [81, 140]. The latter is done to reduce the computational complexity of SVM since it is linearly dependent on the number of training examples and to cope with variable length in the segments. This implies that each $\mathbf{Y}_j^{[o]}$, $\forall j \in J$, is transformed into a single feature vector $\mathbf{y}_j^{[o]}$ containing the mean and standard deviation of each MFCC dimension. The used kernel in the experiments of this chapter is the well-known radial basis function (RBF) and is defined by $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2/2\sigma^2)$ with σ^2 denoting the kernel bandwidth. The SVM hyperparameters, i.e. kernel bandwidth and complexity parameter, are tuned as with GMM by a grid search such that the classification accuracy on the development set is maximised.

3. **Exemplar NMD** (E-NMD): constructs the temporal basis data matrix $\mathbf{A}^{[o]}$ directly from the J labelled examples in the training data $\mathbf{Y}^{[o]}$ instead of learning them as with LRM-NMD. Hence, the training procedure can be omitted but with as main disadvantage that the complexity of the evaluation procedure in E-NMD is linear with the number of exemplars ($L = J$). The unlabelled event is classified by summing the obtained activation vectors of the exemplars corresponding to the same class and the class label $\hat{C}$ is defined by the group of exemplars with the highest accumulated activation.

4. **Supervised NMD** (S-NMD): estimates one dictionary element ($L = 1$) for each class from the isolated class data. This implies that the learning procedure must be run C times. Subsequently, all learned basis matrices are combined into a global temporal basis data matrix $\mathbf{A}^{[o]}$ representing all classes ($L = C$) . Furthermore, it is worth to note that the evaluation and classification procedure of S-NMD is exactly the same as with LRM-NMD.

4.4.3 Single-label learning mode

As already mentioned in Section 4.4.1, single-label learning mode learns the temporal basis data matrix $\mathbf{A}^{[o]}$ from the isolated training events in $\mathbf{Y}_j^{[o]}$, $\forall j \in J$, and the relation between the amount of training data, semantic balancing parameter and sparsity penalty w.r.t. the classification performance will be examined. This experiment is actually a fully supervised setting and is performed in the context to compare LRM-NMD with other supervised learning algorithms such as GMM, SVM and S-NMD. The number of training events per sound class, further denoted by n_{tr}, is artificially increased in the sequence

of $n_{tr} \in \{1, 2, 3, 4, 5, 10, all^1\}$ to investigate its influence on the classification performance of LRM-NMD. The examined semantic balancing parameter values and sparsity penalisation weights are $\eta \in \{0, 0.1, 1, 10, 100, 200, 500, 1000\}$ and $\lambda \in \{0, 5, 10\}$ respectively.

All classification results are listed in Table 7.1 in Appendix 7.2.1 and indicate that the additional weighted sparsity penalty on the activations improves the classification performance of LRM-NMD since the scores for $\lambda = 0$, i.e. no penalisation, are outperformed by the settings $\lambda = 5$ and $\lambda = 10$ respectively. However, too much penalty tends to decreasing classification scores as a result of a sub-optimal factorisation due to the sparseness in $\mathbf{X}$. In this work, the best scores are obtained when $\lambda = 5$, which are also shown in Figure 4.4, and clearly indicate that both the amount of training data and the semantic balancing parameter have a significant impact on the classification performance of LRM-NMD.

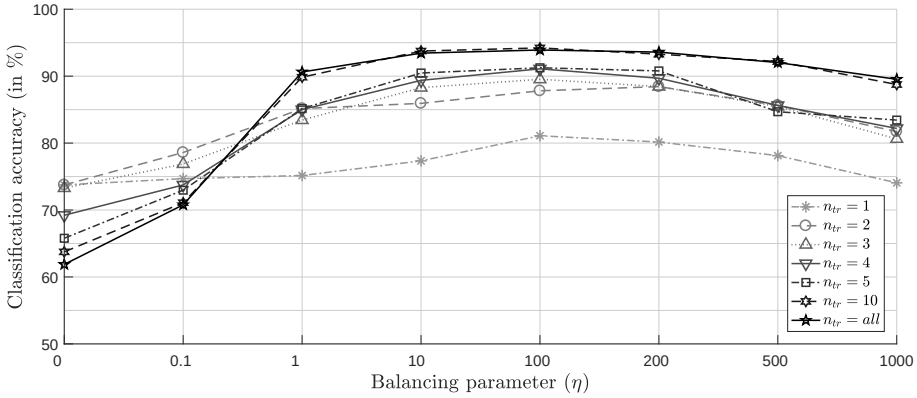

Figure 4.4: Influence of the semantic balancing parameter (η) on the classification performance of LRM-NMD in single-label learning mode for different number of training events per sound class (n_{tr}) when $\lambda = 5$.

First of all, including the labelling information into the learning procedure of LRM-NMD, i.e by setting the semantic balancing parameter $\eta > 0$, boosts the classification performance for all choices of n_{tr} as a result of an improved parts-based representation in $\mathbf{A}^{[o]}$. The highest classification accuracies for most n_{tr} settings are achieved for $\eta = 100$, which are situated between $81.1 \pm 4.2\%$ ($n_{tr} = 1$) and $93.9 \pm 1.2\%$ ($n_{tr} = all$), and remains more or less constant in the range of $\eta \in [10, 200]$. This implies that the choice of η is not critical in this operating mode. Increasing the semantic balancing parameter beyond

[1] $n_{tr} = all$ corresponds to 12 or 13 training examples per class (60% of the class data).

$\eta > 200$ yield decreasing classification scores as a result of overfitting due to too much emphasis on correct factorisation of the labelling information instead of modelling the acoustic event data.

Secondly, increasing the number of training examples per sound class boosts the classification scores as well because of an improved generalisation of the learned temporal basis data $\mathbf{A}^{[o]}$. However, it can be seen this improvement is only realised when the labelling information is included into the learning procedure of LRM-NMD by setting the balancing parameter sufficiently large, i.e. $\eta \geq 1$. Increasing the number of training examples per class without using the labelling information, i.e. situations when $\eta < 1$, leads to less optimal learned dictionary elements. The latter is a result of increased intraclass variabilities within some sound classes due to the higher number of examples making that some sound classes are modelled by two or more dictionary elements while others are not represented at all. This happens because the number of dictionary elements usable for modelling the acoustic event data in the basis data matrix is equal to the number of sound classes since $\mathbf{A}^{[o]}$ is initialised with one example per sound class. Furthermore, the highest gain in classification accuracy in this experiment is achieved when the number of training data per sound class is increased from $n_{tr} = 1$ to $n_{tr} = 2$.

The aspect of the improved parts-based representation and generalisation in $\mathbf{A}^{[o]}$ is visualised in Figure 4.5. This figure shows the initialised and learned temporal basis data of the sound classes *'drawer'*, *'toaster'* and *'locking/unlocking door'* for the parameters $\{n_{tr} = all \ \& \ \eta = 0\}$, $\{n_{tr} = all \ \& \ \eta = 100\}$ and $\{n_{tr} = 1 \ \& \ \eta = 100\}$ when $\lambda = 5$. By comparing the initialised data, i.e. first column, with the learned temporal basis data for $\eta = 0$ and $\eta = 100$ when all training data is used, i.e. second and third column respectively, it can be clearly seen that the balancing parameter η affects the parts-based representation in $\mathbf{A}^{[o]}$ since $\eta = 0$ only tend to model the high energetic spectro-temporal regions while the setting $\eta = 100$ models complete events. The improvement in generalisation on the other hand can be observed by comparing the temporal basis data for $n_{tr} = all$ with the setting of $n_{tr} = 1$ when the balancing parameter remains at $\eta = 100$, i.e. third and fourth column respectively, and can be noticed by smoother learned temporal basis data when all training data is used compared to the case when only one training example is used.

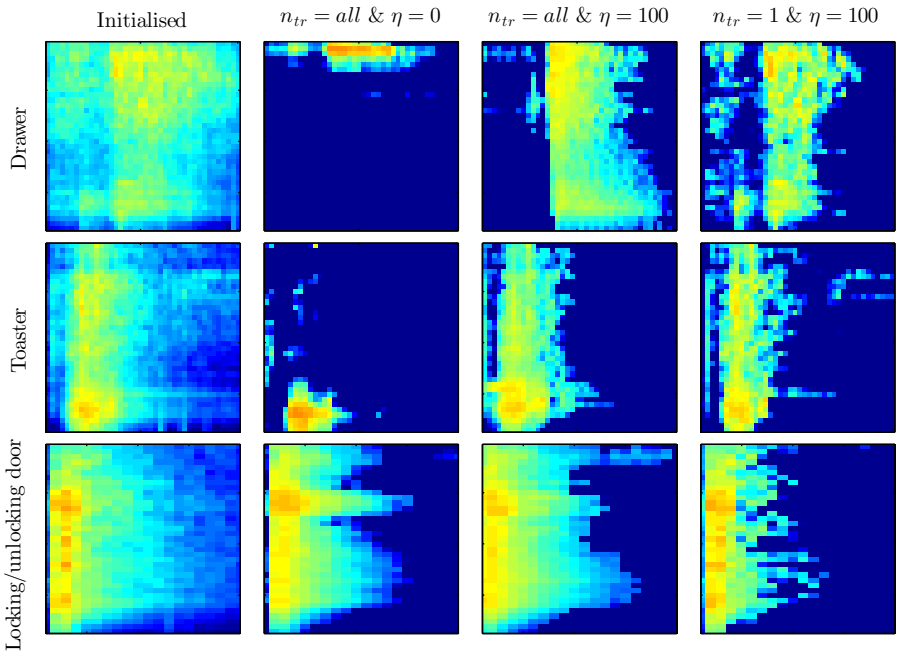

Figure 4.5: A graphical visualisation of the basis data for the classes *'drawer'*, *'toaster'* and *'locking/unlocking door'* for $\lambda = 5$. The leftmost column are the initialised basis data matrices while the next two columns are the obtained data matrices after learning when all training data is used with a semantic balancing parameter of $\eta = 0$ and $\eta = 100$ respectively. The rightmost column are the obtained basis data matrices for $\eta = 100$ when only one training example per sound class is used.

By examining the obtained confusion matrix for the parameter setting $n_{tr} = all$, $\eta = 100$ and $\lambda = 5$, given by Table 4.2, it can be observed that most misclassifications mainly occur within two groups of sound event classes which are defined by {*'fingersnap'* (ID 10), *'handclap'* (ID 11), *'tongue click'* (ID 12)} and {*'locking/unlocking door'* (ID 14),*'opening door'* (ID 15), *'closing door'* (ID 16)}. As one can see, the sound event classes within each group are all related to each other and thereby explaining the confusion. For instance, the first group is linked to human produced sounds while the second group is linked to door sounds. Reducing the number of training events per sound class will decrease the generalisation of the learned dictionary elements and thereby mainly increasing the degree of misclassications within these groups as shown by the confusion matrix for the setting $n_{tr} = 2$, $\eta = 100$ and $\lambda = 5$ given by Table 4.3.

Table 4.2: Confusion matrix of LRM-NMD in single-label learning mode for $\eta = 100$ and $\lambda = 5$ when the number of training events per class is set to $n_{tr} = all$.

Ground truth (ID) \ **Classified label** (ID)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
2	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	-	-	32	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
4	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
6	-	-	-	-	-	**31**	-	-	-	-	-	-	-	-	-	-	-	1	-	-
7	-	-	-	-	-	-	32	-	-	-	-	-	-	-	-	-	-	-	-	-
8	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-
10	-	-	-	-	-	-	2	-	-	**29**	-	1	-	-	-	-	-	-	-	-
11	-	-	-	-	-	-	-	-	-	-	**28**	-	-	2	-	2	-	-	-	-
12	-	-	-	-	-	-	-	-	-	1	4	**21**	3	1	-	2	-	-	-	-
13	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-
14	-	-	1	-	-	-	-	-	-	-	-	-	-	**31**	-	-	-	-	-	-
15	-	-	2	-	-	-	-	-	-	-	-	-	-	3	**20**	6	-	-	-	-
16	-	-	-	-	-	-	-	-	-	-	-	1	-	4	1	**26**	-	1	-	-
17	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-	-
18	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-
19	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	**31**	-
20	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**32**

Note: see Table 4.1 for the class names.

Table 4.3: Confusion matrix of LRM-NMD in single-label learning mode for $\eta = 100$ and $\lambda = 5$ when the number of training events per class is set to $n_{tr} = 2$.

Ground truth (ID) \ **Classified label** (ID)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
2	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	-	-	**31**	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
4	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5	-	-	7	-	**24**	-	1	-	-	-	-	-	-	-	-	-	-	-	-	-
6	-	-	-	-	-	**31**	1	-	-	-	-	-	-	-	-	-	-	-	-	-
7	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-
8	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-	-	**31**	-	-	-	-	-	-	-	1	-	-	-
10	-	-	-	-	-	-	2	-	-	**28**	-	-	-	1	-	-	-	1	-	-
11	-	-	4	-	-	-	-	-	-	-	**25**	1	-	-	-	2	-	-	-	-
12	-	-	2	-	-	-	-	-	-	3	3	**16**	3	3	-	2	-	-	-	-
13	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-
14	-	-	3	-	-	-	-	-	-	-	1	-	-	**26**	1	1	-	-	-	-
15	-	-	2	-	-	-	-	-	-	1	1	-	-	-	**24**	4	-	-	-	-
16	-	-	1	1	-	-	-	-	-	-	-	-	-	7	4	**17**	-	2	-	-
17	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-	-
18	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-
19	-	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	**27**	-
20	-	-	1	2	-	1	-	-	-	-	-	-	1	-	-	-	-	-	1	**26**

Note: see Table 4.1 for the class names.

By comparing the classification scores of LRM-NMD when $\eta = 100$ and $\lambda = 5$ with the baseline results in function of the amount of training data, which are shown in Figure 4.6, it can be clearly seen that LRM-NMD outperforms the baselines, with exception of S-NMD, when the available number of training events per sound class is rather limited, i.e. $n_{tr} = 5$ or less. However, the main advantage of LRM-NMD over S-NMD is that it can be used in a low-resolution multi-label learning mode, i.e. larger segments are labelled by a single multi-label vector indicating sound classes of the events, or possibly a subset of events, that take place during that period, and thereby reducing the workload on labelling the training data significantly as will be discussed in Section 4.4.4. The best sparsity penalty for S-NMD and E-NMD is also $\lambda = 5$. The highest classification scores when all training data is used are obtained by GMM followed by SVM, S-NMD and LRM-NMD respectively. Furthermore, learning the temporal basis data $\mathbf{A}^{[o]}$ from the training data is preferred over an exemplar based approach since the E-NMD results are always exceeded by both LRM-NMD and S-NMD for all n_{tr} settings.

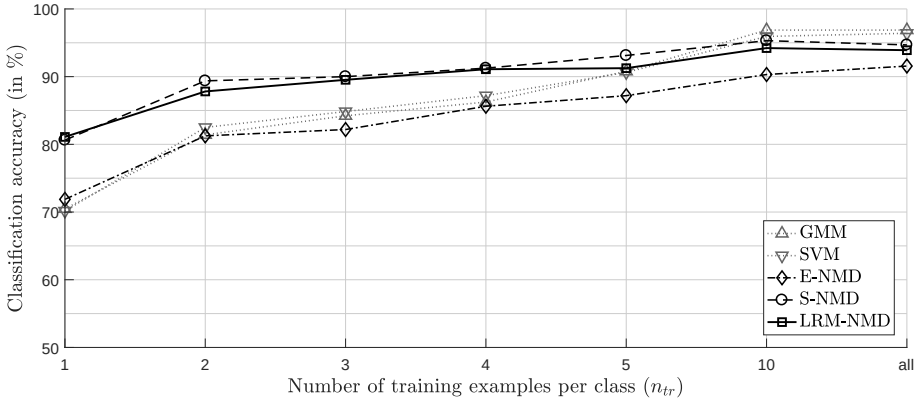

Figure 4.6: Classification scores of LRM-NMD in single-label learning mode for $\eta = 100$ and $\lambda = 5$ together with the obtained baseline results in function of different number of training examples per sound class (n_{tr}).

The robustness to environmental background noise of LRM-NMD when $\eta = 100$ and $\lambda = 5$ is shown in Figure 4.7. These results clearly indicate that LRM-NMD can deal with noise levels down to 10 dB SNR without any significant loss in classification performance. Lower SNRs yield decreasing classification scores, however the absolute reduction in classification performance from 10 dB to 0 dB strongly depends on the amount of training data, i.e. less than 10% in case of $n_{tr} = \{10, all\}$ and somewhere around 20% to 25% in case of $n_{tr} = \{1, 2\}$. The parameter settings of $n_{tr} = \{3, 4, 5\}$ achieve still promising classification scores for a SNR of 0 dB and are situated in the range of 75% and 80% respectively.

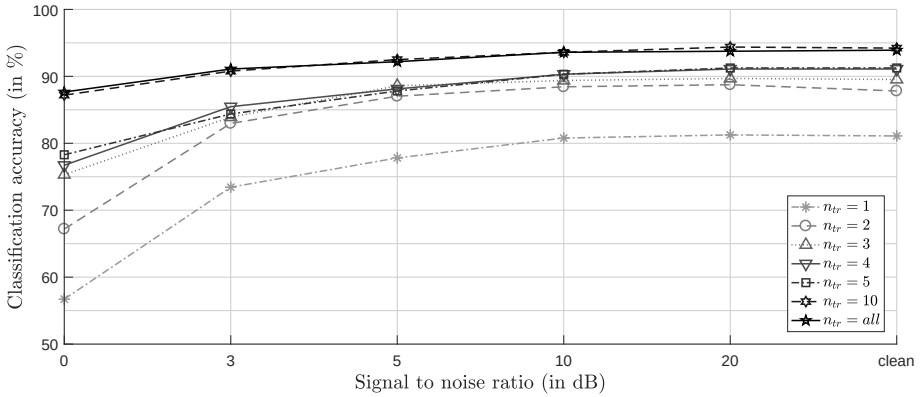

Figure 4.7: Background noise robustness of LRM-NMD in single-label learning mode when $\eta = 100$ and $\lambda = 5$ in function the signal-to-noise ratio (SNR) for different number of training examples per sound class (n_{tr}).

4.4.4 Low-resolution multi-label learning mode

As already mentioned, low-resolution multi-label learning mode learns the temporal basis data $\mathbf{A}^{[o]}$ directly from segments spanning several events instead of isolated event data as with single label learning from Section 4.4.3. Hence, each $\mathbf{Y}_j^{[o]}$, $\forall j \in J$, is an segment labelled by $\mathbf{y}_j^{[s]}$ indicating the sound classes, or a subset of sound classes, of the events that take place. This approach of annotating the data reduces the labelling workload significantly since it is no longer needed to segment the data on the event level by indicating the onset and offset times. This makes that the low-resolution multi-label learning mode is of more interest in practical settings, e.g. home monitoring, where typically large datasets need to be collected to learn the model specific parameters.

In this experiment, each segment is made up of five events by randomly combining the Frobenius normalised events from the training set. The events in the segments are separated by T frames containing small positive values uniformly distributed between the range ϵ (floating-point relative accuracy) and 10^{-4} to mimic short periods of silence between two successive events. In total 48 annotating segments are created per fold since each fold is represented by 240 isolated training examples which are allowed to be used once. The minimum and maximum time duration of the segments varies from 1.86 to 6.41 seconds. In this experiment, the influence of the semantic balancing parameter (η) together with the number of labelled events per segment, further denoted by n_{lbl}, will be examined. For instance, $n_{lbl} = 1$ yield only one labelled event out

of five events in each segment. Furthermore, the sound events in the segments are sampled in such a way that they all have a different sound class combination and that the amount of labelled data per sound class is balanced for all choices of $n_{lbl} \in [1, 5]$.

In addition, the influence of overlapping events in the segments on the classification performance of LRM-NMD will be investigated as well and in the range of $n_{overlap} \in \{0\%, 25\%, 50\%, 75\%, 100\%\}$. Important to note is that the events in the segments are sorted with increasing time duration and that $n_{overlap}$ is defined as the percentage of overlapping frames between two successive events computed on basis of the shortest one as shown in Figure 4.8 (b). Special cases are $n_{overlap} = 0\%$ where the events are still separated with T frames containing small positive random values, as illustrated in Figure 4.8 (a), and $n_{overlap} = 100\%$ where all five events have the same onset time, as illustrated in 4.8 (c).

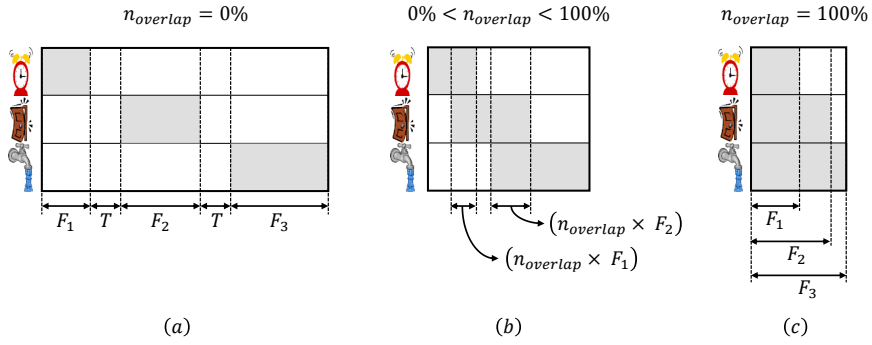

Figure 4.8: Graphical visualisation of the definition $n_{overlap}$. Situation (a) corresponds to the situation $n_{overlap} = 0\%$ where the events are all separated by T frames. Situation (b) to $0\% < n_{overlap} < 100\%$ and the amount of overlapping frames is defined in percentage w.r.t. the shortest one. Situation (c) corresponds to $n_{overlap} = 100\%$ where all events have the same onset time.

All classification results on the clean and non-overlapping data are listed in Table 7.2 in Appendix 7.2.2 and the best classification scores are again when the sparsity penalty is set to $\lambda = 5$. The corresponding results are shown in Figure 4.9 and indicate that the optimal choice of semantic balancing parameter in this learning mode depends on the number of labelled events per segment since increasing η tends to earlier classification performance breakoffs for lower n_{lbl} settings. The latter is a result of the penalty induced by the semantic balancing parameter η on modelling the non-labelled event data in the segments

$\mathbf{Y}_j^{[o]}$, using their temporal basis matrices in $\mathbf{A}^{[o]}$, which leads to an increased divergence between $\mathbf{y}_j^{[s]}$ and $\boldsymbol{\psi}_j^{[s]}$. In this experiment, a good choice of semantic balancing parameter is $\eta = 5$ as it achieves high classification scores for all n_{lbl} settings, i.e. around $73.9 \pm 5.3\%$ for $n_{lbl} = 1$ and $90.2 \pm 3.9\%$ for $n_{lbl} = 5$, and will thus be used during the next experiments.

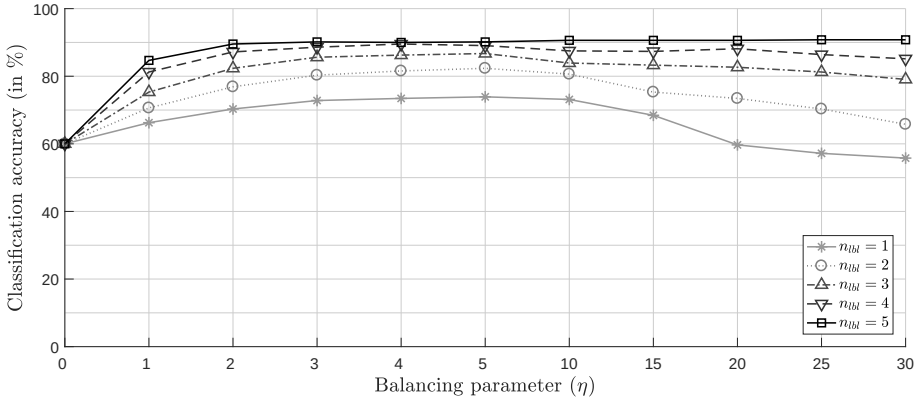

Figure 4.9: Influence of the semantic balancing parameter (η) on the classification performance of LRM-NMD in low-resolution multi-label learning mode in function of different number of labelled events per segment (n_{lbl}) when $\lambda = 5$ and $n_{overlap} = 0\%$.

The obtained confusion matrix when all events are labelled in each segment ($n_{lbl} = 5$) is given by Table 4.4 and indicates the same trends as with single-label learning. More specifically, most misclassifications occur again within the sound groups {*'fingersnap'* (ID 10), *'handclap'* (ID 11), *'tongue click'* (ID 12)} and {*'locking/unlocking door'* (ID 14), *'opening door'* (ID 15), *'closing door'* (ID 16)}. Reducing the number of labelled events per segment mainly increases the number of misclassifications within these sound classes. The latter can be observed in the confusion matrix when only two out of five events are labelled, i.e. $n_{lbl} = 2$, which is given by Table 4.5.

Table 4.4: Confusion matrix of LRM-NMD in multi-label learning mode for $\eta = 5$ and $\lambda = 5$ when all events are labelled in each segment ($n_{lbl} = 5$).

Ground truth (ID) \ **Classified label** (ID)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
2	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	-	-	**27**	4	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
4	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
6	-	-	-	-	-	**30**	-	-	-	1	-	-	-	-	-	-	-	1	-	-
7	-	-	-	2	-	-	**29**	-	-	-	-	-	-	-	-	-	-	1	-	-
8	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-
10	-	-	-	-	-	-	-	-	-	**27**	-	2	-	1	-	-	-	2	-	-
11	-	-	-	-	-	-	-	-	-	-	**27**	-	-	3	2	-	-	-	-	-
12	-	-	-	-	-	-	-	-	-	2	4	**19**	4	2	1	-	-	-	-	-
13	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-
14	-	-	-	1	-	-	-	-	-	-	2	-	-	**29**	-	-	-	-	-	-
15	-	-	-	-	-	-	-	-	-	1	2	-	-	6	**23**	-	-	-	-	-
16	-	-	-	-	-	-	-	-	-	-	-	-	-	4	1	**27**	-	-	-	-
17	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-	-
18	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-
19	-	-	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	5	**25**	-
20	-	-	3	2	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	**26**

Note: see Table 4.1 for the class names.

Table 4.5: Confusion matrix of LRM-NMD in multi-label learning mode for $\eta = 5$ and $\lambda = 5$ when two out of five events are labelled in each segment ($n_{lbl} = 2$).

Ground truth (ID) \ **Classified label** (ID)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
2	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	-	-	**17**	13	-	-	-	-	-	-	2	-	-	-	-	-	-	-	-	-
4	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
6	-	-	-	-	-	**30**	-	-	-	2	-	-	-	-	-	-	-	-	-	-
7	-	-	-	-	-	-	**31**	-	-	-	-	-	-	-	-	-	-	1	-	-
8	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-	-	-	-	-
10	-	-	-	-	-	1	-	-	-	**26**	1	1	-	-	1	-	-	2	-	-
11	-	-	1	-	1	-	-	-	-	1	**22**	-	-	7	-	-	-	-	-	-
12	-	-	-	-	-	-	-	-	-	2	5	**18**	4	3	-	-	-	-	-	-
13	-	-	-	-	-	-	-	-	-	-	-	-	**32**	-	-	-	-	-	-	-
14	-	-	-	3	-	-	-	-	-	-	3	-	-	**26**	-	-	-	-	-	-
15	-	-	-	-	-	-	-	-	-	1	1	-	-	7	**23**	-	-	-	-	-
16	-	-	-	-	-	-	-	-	-	-	-	-	1	13	1	**17**	-	-	-	-
17	-	-	4	-	-	-	-	-	-	-	-	-	-	-	-	-	**28**	-	-	-
18	-	-	-	-	-	-	6	-	-	-	-	-	-	-	-	-	-	**23**	3	-
19	-	-	-	4	-	-	-	-	-	-	-	-	-	-	-	-	-	11	**17**	-
20	-	-	2	4	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	**25**

Note: see Table 4.1 for the class names.

By comparing the classification scores of LRM-NMD for $\eta = 5$ and $\lambda = 5$ with the results obtained by GMM, SVM and S-NMD in function of n_{lbl}, which are shown in Figure 4.10, it can be clearly seen that LRM-NMD is capable of dealing with low-resolution multi-label training data. It is worth to note that the sparsity penalty for S-NMD is also set to $\lambda = 5$ and that E-NMD is not evaluated in this learning mode since the isolated exemplars are not available. Furthermore, the sound class model parameters of GMM, SVM and S-NMD are estimated from the segments containing at least one labelled example of the corresponding sound class. This implies that the training data contains acoustic information of other sound classes as well and thereby resulting into less accurate model parameter estimation which explains the decreasing classification accuracies. In addition, it can be seen that GMM is also able to deal with low-resolution multi-label training data, as a result of the ability to model different regions in the feature space and thereby distinguishing the different event classes, but that the classification performance is strongly related to the number of labelled examples.

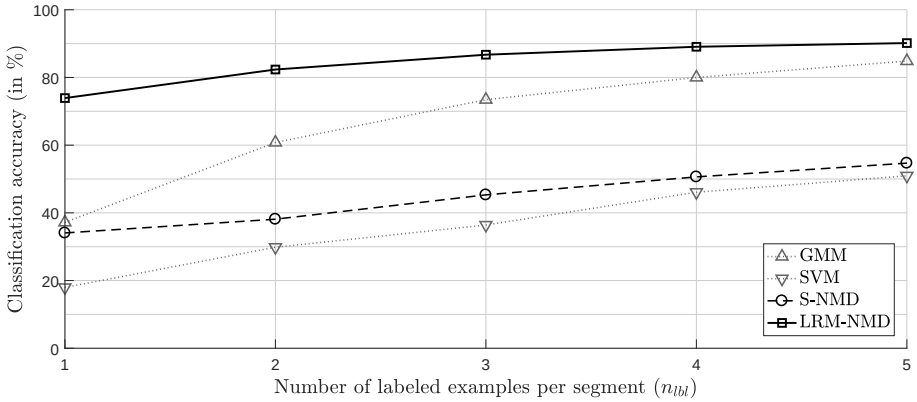

Figure 4.10: Classification scores of LRM-NMD in low-resolution multi-label learning mode on the non-overlapping data ($n_{overlap} = 0\%$) for $\eta = 5$ and $\lambda = 5$ together with the obtained baseline results in function of different number of labelled events per segment (n_{lbl}).

Figure 4.11 are the classification results of LRM-NMD in noisy conditions and indicate that, as with single-label learning, LRM-NMD in low-resolution multi-label learning mode can deal with noise levels up to 10 dB without any significant loss in classification performance. Lowering the SNR further decreases classification scores. The absolute reduction in classification performance from 10 dB to 0 dB is situated somewhere between 30% and 35% for all n_{lbl} settings.

This implies that the number of labelled events per segment has a limited influence on the noise robustness of LRM-NMD since the amount of training data remains the same for all n_{lbl} settings in this operating mode. Furthermore, it can be observed that the highest drop in classification performance is when the SNR is lowered from 3 dB till 0 dB and that an SNR of 3 dB achieves still classification scores in the range of 70% to 85% as long as $n_{lbl} \geq 2$.

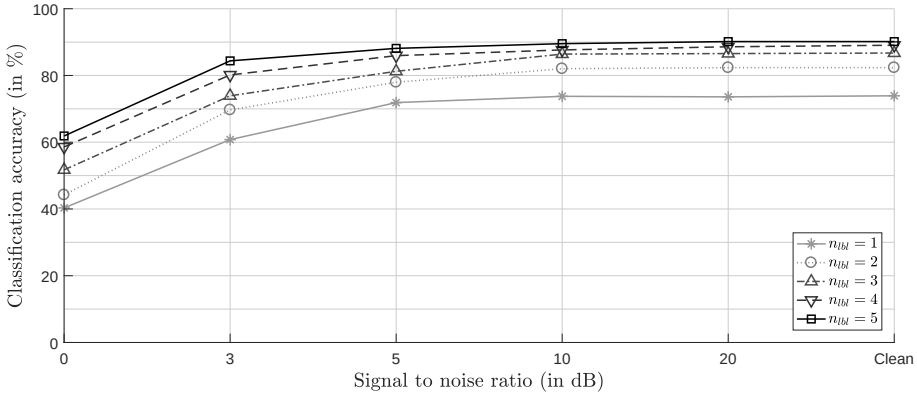

Figure 4.11: Background noise robustness of LRM-NMD in low-resolution multi-label learning mode when $\eta = 5$ and $\lambda = 5$ in function of the signal to noise ratio (SNR) for different number of labelled events per segment (n_{lbl}).

The influence of overlapping events in the segments on the classification performance of LRM-NMD is shown in Figure 4.12 where the obtained classification results on the clean data are visualised in function of $n_{overlap}$. As expected, increasing $n_{overlap}$ yields decreasing classification scores, however the drop in classification performance from $n_{overlap} = 0\%$ to $n_{overlap} = 50\%$ is rather limited for all n_{lbl} settings and is situated around 5%. Increasing the amount of overlap further results in higher decreasing classification scores. The obtained classification results on the noisy data in function of $n_{overlap}$ are all listed in Table 7.3 in Appendix 4.4.4 and indicate the same trends as with the clean data.

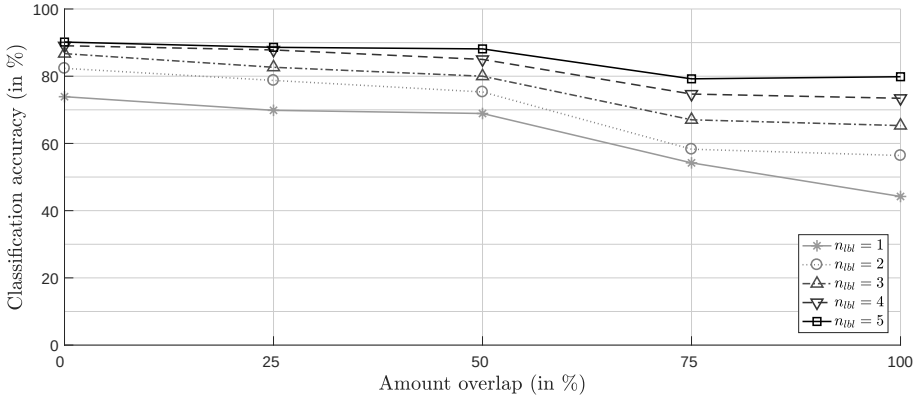

Figure 4.12: Classification scores of LRM-NMD in low-resolution multi-label learning mode for $\eta = 5$ and $\lambda = 5$ in function of the amount of overlap within the training events ($n_{overlap}$) on clean data for different number of labelled events per segment (n_{lbl}).

4.5 Discussion

During all experiments in the previous section, each sound class was represented by a single dictionary element in the three-way tensor $\mathbf{A}^{[o]}$ since the degree of variability within the examples of each sound class was rather limited. This makes that one dictionary element per sound class in $\mathbf{A}^{[o]}$ was sufficient for modelling the data. In case of significant intra-class variabilities, e.g. door knock on a wooden door versus a composite door, it is perfectly possible using multiple dictionary elements in $\mathbf{A}^{[o]}$ for modelling the same sound class. However, choosing a 'correct' number of dictionary elements per sound class in advance is not that convenient and is strongly related to the data and setting. Strategies are available in the literature to tune the number of dictionary elements from the data and is achieved via automatic relevance determination (ARD), a technique that has been employed in Bayesian Principal Component Analysis (PCA) and sparse Bayesian learning [165, 135]. The same ARD methodology can be applied to LRM-NMD for tuning the number of dictionary components per sound class from the data.

Regarding the previous paragraph, it is also expected that in real-life settings an on-line learning strategy will be of more interest than an off-line (batch) setup. This implies that (1) both the amount of dictionaries in $\mathbf{A}^{[o]}$ and the number of sound classes in $\mathbf{A}^{[s]}$ can change over time (i.e. when a new sound class has been occurred in the data) and (2) that the learned dictionaries are allowed to adapt to the current situation (e.g. when an appliance has been

replaced by another one producing a different type of sound). A possible on-line learning strategy is that after each period of time the learned dictionaries in $\mathbf{A}^{[o]}$ are updated if necessary given the recently recorded data. In case of newly collected sound classes in the data, additional dictionary elements must be added to $\mathbf{A}^{[o]}$ such that these sound classes are represented as well. The latter can be done by adding one or more isolated examples of each new sound class to $\mathbf{A}^{[o]}$ and linking them as such given the labelling matrix $\mathbf{A}^{[s]}$. Next, both the set of dictionary elements in $\mathbf{A}^{[o]}$ and the labelling matrix $\mathbf{A}^{[s]}$ can be updated from the newly acquired data and the low-resolution mulit-labelling information using the LRM-NMD learning procedure given in Algorithm 1. Incremental and adaptive learning in the domain of non-negative matrix factorisation is already widely examined and successfully applied in the field of structure discovery [18, 40, 186]. The same on-line learning strategies can also be applied to LRM-NMD.

Learning a classifier on top of the activations is also an interesting aspect that will be investigated. During the experiments of this chapter, the predicted class label $\hat{C}$ is simply defined by the temporal basis matrix $\mathbf{A}_{(l)}$ with the highest accumulated activation. However, the work in [14, 13, 15] proposes a task-driven non-negative matrix factorisation (TNMF) based approach which merges the dictionary learning and the training of the classifier in a joint optimisation problem. Hence, the objective of TNMF is to minimise both the reconstruction error and the classification cost in a common problem. Influenced by the classifier, the dictionary of the TNMF model is encouraged to explain the discriminative information in the data while keeping a low reconstruction cost and thereby boosting the classification problem. A similar strategy can also be applied to LRM-NMD.

Another interesting aspect of LRM-NMD is that it can also deal with data labelled on the activity level instead of the event level as with the experiments in the previous section. An activity can be seen as a sequence of potentially overlapping isolated sound events and each activity is characterised by its own specific combination of sound events. For instance, *'brushing teeth'* contains data from the sound class *'tap water'* in combination with the sound of a *'tooth brush'* while the activity *'shaving'* contains also data of *'tap water'* but this time in combination with the sound of a *'razor'*. This implies that some dictionary elements in $\mathbf{A}^{[o]}$ must be linked to multiple activities while others are simply linked to one specific activity. The latter can be done by initialising both the three-way tensor $\mathbf{A}^{[o]}$ and the labelling matrix $\mathbf{A}^{[s]}$ randomly with positive values, and let LRM-NMD learn the set of dictionary elements in $\mathbf{A}^{[o]}$, i.e. the isolated sound events that take place in the data, and to which activity they can be linked given the labelling matrix $\mathbf{A}^{[s]}$. The overall decision of which activity has been occurred is done over a longer period of time by accumulating

the activations of the activity related dictionaries. The activity with the highest degree can be assumed as the one occurring in the data.

4.6 Conclusion

This chapter focusses on modelling sound events from acoustic data labelled on a low-resolution and multi-label level using non-negative matrix deconvolution. The low-resolution multi-labelling information simply indicates the sound classes of the events that take place over a longer period of time in the acoustic data without identifying beginning nor endings of the individual events. This approach of labelling the data requires less annotation work compared to protocols where the data must be labelled on an event level. In order to cope with this type of labelling, we have extended the existing NMD algorithm into a low-resolution multi-label variant named LRM-NMD by incorporating the low-resolution multi-labelling information into the learning procedure of NMD.

Firstly, we have examined LRM-NMD in a single-label learning context, i.e. all events are labelled individually, to benchmark our proposed algorithm with other already widely examined sound event classifiers such as GMM, SVM, S-NMD and E-NMD [76, 77, 159, 176]. The obtained classification results show that LRM-NMD, together with S-NMD, outperforms GMM, SVM and E-NMD in situations when the amount of training examples per sound class is rather limited, i.e. $n_{tr} \leq 5$, and are situated in the range of 81% and 91% when $n_{tr} = 1$ and $n_{tr} = 5$ respectively. However, the main advantage of LRM-NMD over S-NMD is that it can be used in a low-resolution multi-label setting which is of more interest in real-life settings. In addition, LRM-NMD is also able to deal directly with overlapping event data without any further modifications to the algorithm due to its additive property, which is an interesting property in real-life settings as well, while this is not the case for GMM and SVM. It is also shown that learning the temporal basis data from the acoustic data is preferred over an exemplar based approach since E-NMD is always outperformed by both LRM-NMD and S-NMD. Furthermore, the robustness of LRM-NMD to stationary environmental background noise has been shown till SNRs of 10 dB without any significant loss in classification performance. Lowering the SNRs further yield decreasing classification scores, but the absolute reduction in classification performance is strongly related to the number of training examples per sound class.

Secondly, we have shown that LRM-NMD is capable of modelling sound events directly from acoustic data labelled on a low-resolution and multi-label level. In this work, the segments are all constructed out of five events and the relation between the number of labelled events per segment (n_{lbl}) w.r.t. the classification performance of LRM-NMD is examined. The corresponding

classification accuracies are situated in the range of 75% and 90% when $n_{lbl} = 1$ and $n_{lbl} = 5$ respectively and outperform the baselines. Although GMM is able to achieve a classification accuracy slightly below LRM-NMD when all events in the segments are labelled, it rapidly decreases to an accuracy of less than 40% when $n_{lbl} = 1$ and thereby making it less usable compared to LRM-NMD. The robustness to background noise has been shown again with SNRs of down to 10 dB without any significant decrease in classification performance. We have also shown that LRM-NMD can deal with overlapping events in the segments as well, i.e. a drop in absolute classification performance of somewhere around 5% when $n_{overlap} = 50\%$. Further increasing $n_{overlap}$ yields larger declines in classification scores as a result of a higher complexity in the acoustic observation data.

Chapter 5

On-edge adaptive acoustic presence detection

This chapter is devoted to the development of an on-edge and adaptive acoustic presence detection classifier. The focus is situated in using a simple and resource efficient machine learning algorithm which can be mapped to devices with limited computational resources. The adaptive learning aspect of the classifier yields that the model parameters are continuously updated locally from the recently collected samples only. Hence, no data must be stored for learning the model parameters and thereby reducing the required memory footprint on the embedded platform. During the experiments we will present two different adaptive learning strategies based on a least-squares support vector machine (LS-SVM) framework and will show that similar trends in the results can be obtained as with offline learning.

The work presented in this chapter is submitted as an internal written report:

- Vuegen, L. and Karsmakers, P. "On-edge adaptive acoustic presence detection." In final VLAIO report for the O&O project: Research towards a Perceptive Sensing Home Control System (PERSES).

5.1 Introduction

Recent advances in the domain of '*Internet of Things*' (IoT) have led to a wide range of off-the-shelf connected devices especially developed for (indoor) monitoring applications [153, 154]. Today's IoT devices are known to be small, computationally powerful, energy efficient and rather cheap allowing unobtrusive integrations in domestic and public environments at a low cost. As a result, the use of IoT-enabled devices to obtain an intelligent behaviour and to facilitate home automation gained a lot of research interest recent years and is known by '*smart homes*' and '*smart environments*' [1, 23, 158]. The main difference between smart homes and traditional home automation systems, i.e. domotics, is situated in the ability of sensing the environment. Being able to sense the environment allows the system to learn the normal behaviour of the residents/users and to adapt to their specific lifestyles. Ideally, a smart environment knows who you are, where you are and what you want, all without having to tell it anything. For instance, one of today's most well-known smart home applications are the smart thermostats which are able to teach themselves from user provided inputs.

Research has shown that residential, public and commercial buildings account for 30% to 40% of the total energy demand in the developed countries [21, 34, 133]. The main energy consumers are the so-called lighting, heating, ventilation and air conditioning (L-HVAC) systems accounting for about 70% of the total energy consumption [121, 133]. Reliable presence detection could dynamically control the L-HVAC devices resulting in a reduced energy consumption and an improved user comfort. Passive infrared (PIR) sensors are currently the most commonly used type of presence (motion) detectors due to their accurate detection performance, low power requirements, and are easy to install and calibrate [43, 64, 184]. However, the main disadvantages of PIR based presence detection are (i) the need of line-of-sight (LOS) path, (ii) limited operating range and field of view (FOV), (iii) the inability to detect static persons, and (iv) that it cannot provide detailed context information about the environment being monitored [91]. As a result, accurate presence detection using context aware sensors gained a lot of research interest recent years.

The authors of [20, 114, 167] propose an ultrasound based presence detection system where short bursts of pulses are transmitted and received by an ultrasonic array. Information about occupancy and location is obtained through changes in the echo intensity of the transmitted signal. In [152] the authors propose an ultrasound based presence detection system with person identification by measuring the persons height. The average detection range of ultrasound is comparable to that of a PIR sensor (approx. 10 meters), but it can also detect near static changes and even motions behind partial obstructions [68]. However,

the high sensitivity of ultrasonic sensors makes them also susceptible to false detections often triggered by airflows or vibrations [64, 91]. Furthermore, some domesticated animal species (e.g. dogs, cats, parakeets, guinea pigs) have an auditory range far within the ultrasound spectrum [155]. As a result, the use of an ultrasound based presence detection system is not recommended in these situations. The authors of [9, 42, 107] propose a video based presence detection system. Image recordings have been demonstrated to provide accurate information on location [52], count [29], activity [141], identity [180] and track [46], but it comes with the cost of advanced signal processing and expensive hardware. In addition, video images are highly affected by changes in background illumination, require a line-of-sight operation (LOS), and are often considered as privacy intrusive [31]. A radar based presence detection system is proposed in [17, 25, 182]. Like ultrasound, transmitting radio-frequency (RF) signals can provide information about occupancy and location by means of analysing the echoes. The high-frequency component enables non-line-of-sight (NLOS) operations, and even detections through solid objects, but it comes with the cost of expensive and energy consuming hardware [134]. The use of environmental sensors (e.g. air temperature, humidity, air pressure, carbon dioxide, etc.) and utility sensors (e.g. water, gas, electricity, etc.) gained also attention in the literature [84, 86, 93, 112]. These sensors are known to be cheap, privacy unintrusive and easy to process (low data rates) but cannot provide detailed context about the environment. As a result, environmental and utility sensors are typically used in combination with more descriptive sensors to further improve the overall detection performance [84].

Although acoustics are a rich source of information that glean useful insights about the monitoring context, the use of acoustics as a sensing modality for occupancy detection received limited attention in the research community. The proposed acoustic presence detection approaches in the literature are either focussing on classifying daily activities [117, 123] or in classifying human produced sounds such as speech [28, 73, 72] and footsteps [75, 143, 174]. The development of an acoustic occupancy detector being able to discriminate '*presence*' from '*absence*' related sounds is to the best of our knowledge not yet investigated. The work proposed in this chapter will mainly concentrate on the development of an '*adaptive*' and '*resource efficient*' acoustic presence detection algorithm which can be embedded locally on devices with limited computational resources. The aspect of adaptive learning implies that the model parameters can be updated in a stepwise manner from small batches of newly arrived samples only. The latter reduces the required memory footprint of the embedded platform and allows to model slowly varying changes in the data as well which is an desirable property in case of real-life settings. For instance, moving or replacing the furniture in a home environment can change the room acoustics and thereby affect the learned classifier, seasonal changes may result

in other types of background noise which are not yet modelled by the classifier, and so on. In addition, local featurisation, modelling and classification at sensor side reduces the required communication bandwidth, i.e. no (featurised) data must be transmitted, and improves both user privacy and security. Embedding machine learning next to the sensor (instead of the gateway or cloud) is called in this work by '*machine learning on the extreme edge*'.

The remainder of this chapter is organised as followed: Section 5.2 is devoted to the proposed system architecture for the purpose of acoustic presence detection. Section 5.3 introduces the on-edge presence detection classifiers which can be embedded locally on devices with limited resources. The adaptive learning strategy is explained in Section 5.4. The experimental specific details will be discussed in Section 5.5 and the obtained results are given in Section 5.6. Finally, the discussion and final concluding remarks are given in Section 5.7 and Section 5.8 respectively.

5.2 System architecture

The proposed system architecture is shown in Figure 5.1 and consists of three stages. The first stage is the acoustic front-end where the captured sound data is pre-amplified and A/D-converted. The second stage is the acoustic pre-processing stage where the data is framed into short overlapping frames. Each frame is then evaluated by an energy based sound activity detector (SAD) and only the selected frames will be considered as acoustically relevant and will be further processed. The last stage is the local classifier for presence detection which will learn its parameters in an '*adaptive*' and '*resource efficient*' fashion from the newly collected SAD relevant data samples.

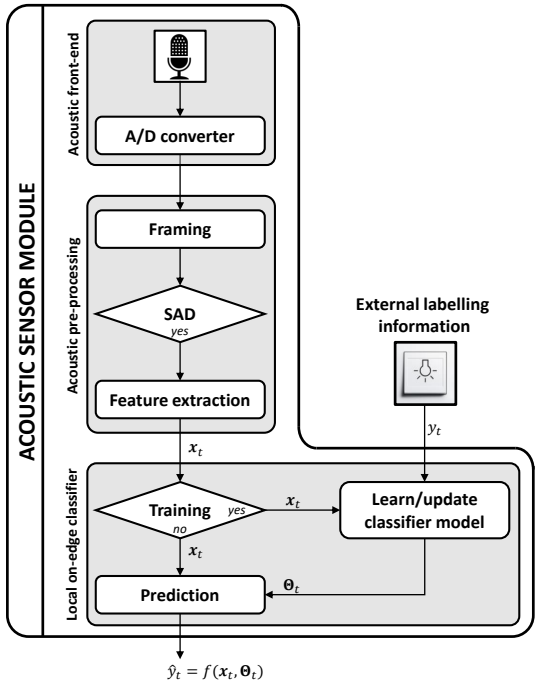

Figure 5.1: Graphical visualisation of the used system architecture. The architecture is composed of an acoustic front-end, an acoustic pre-processing stage and a local on-edge presence detection classifier.

5.2.2.1 Acoustic front-end

The acoustic front-end consists of a miniature, high-performance, low power, analog MEMS microphone in combination with a low power stereo A/D-converter especially developed for wireless and portable audio applications. The used microphone is the Knowles SiSonic SPH1611LR5H-1 which is characterised by a flat frequency response in the audio range and an omnidirectional sensitivity pattern [89]. The used A/D-converter is the Texas Instruments TLV320ADC3101 with embedded miniDSP featuring an automatic gain control (AGC) [170]. The used sampling frequency was set to 32 kHz with a 16 bit A/D-resolution and enabled AGC.

5.2.2.2 Acoustic pre-processing

The pre-processing stage transforms the raw acoustic data into short overlapping frames of 25 ms with a frame shift of 10 ms. Each frame is then evaluated by the SAD to determine whether it contains sufficient energy for further processing or not. The average energy E_t of a frame $\mathbf{s}_t$ is computed by using

$$E_t = \frac{1}{N} \sum_{n=1}^{N} \mathbf{s}_t(n)^2, \tag{5.1}$$

with $\mathbf{s}_t(n)$ denoting the nth-sample in frame $\mathbf{s}_t$. The corresponding frame is considered as acoustically relevant if the average energy E_t exceeds a predefined threshold which was tuned manually for the office environment. Only the acoustically relevant frames will be further processed and thereby saving computational resources. However, it is worth to note that a hangover timer of 500 ms (i.e. 50 frames) is added to prevent rapid toggling of the SAD detections.

The SAD detected frames are then transformed into the MFCC representation as discussed in Section 2.2.1. The MFCCs are further processed into a statistical representation containing the cepstral means and standard deviations to reduce the number of feature vectors per time instance. The latter is done on a 500 ms window basis (50 frames) with an overlap of 250 ms (25 frames). Hence, each window is represented by a single feature vector $\mathbf{x} = [\mu_1, \sigma_1, \mu_2, \sigma_2, \ldots, \mu_d, \sigma_d, \ldots, \mu_D, \sigma_D]$ of length $2D$ with μ_d and σ_d denoting the dth-dimensional cepstral mean and standard deviation respectively.

Table 5.1 gives an overview of the computational complexity of the pre-processing stage in terms of the number of operations per arithmetic operator together with the required memory to store the feature extraction parameters and the processed data. The required memory is expressed in bytes and depends on the precision (in bytes) of the used data type, e.g. boolean, integer, floating-point, etc., and is denoted by S. The division per arithmetic operator gives a more elaborate insight of the overall complexity due to differences in cost per operator [36]. For instance, multiplying two $n \times n$ matrices involves $\mathcal{O}(n^2)$ operations while inverting a $n \times n$ matrix typically yields $\mathcal{O}(n^3)$ operations.

Table 5.1: Overview of the computational complexity and the required memory allocation of the pre-processing stage.

Stage	**Operation**		**Parameter storage** (in bytes)	**Data storage** (in bytes)
SAD	mac:	N	$1 \cdot S$	$N \cdot S$
	mult:	1		
	comp:	1		
WIN	mult:	N	$N \cdot S$	$N \cdot S$
FFT	c-mac:	$K \cdot \log_2(K)$	$K \cdot S$	$(1/2 \cdot K) \cdot S$
	abs:	$1/2 \cdot K$		
LOGMEL	mac:	$1/2 \cdot K \cdot B$	$(1/2 \cdot K \cdot B) \cdot S$	$B \cdot S$
	log:	B		
DCT	mac:	$B \cdot D$	$(B \cdot D) \cdot S$	$(F \cdot D) \cdot S$
$\mu\sigma$-MFCC	add:	$3 \cdot D \cdot F$	-	$(2 \cdot D) \cdot S$
	mult:	$(2 + F) \cdot D$		
	root:	D		

Stage abbreviations: SAD: sound activity detector, WIN: windowing, FFT: fast Fourier transformation, LOGMEL: log-transformed MEL-magnitude features, DCT: discrete cosine transformation and $\mu\sigma$-MFCC: mean and standard deviation per MEL-frequency cepstral coefficient.
Operation abbreviations: abs: absolute value, add: addition, comp: comparator, c-mac: complex multiply-accumulate, log: natural logarithm, mac: multiply-accumulate, mult: multiplication and root: square root.
Notations: N: number of samples per window, K: number of frequency bins, B: number of band in the MEL-filter bank, D: number of cepstral coefficients, F: number of frames and S: data type precision.
Note: The computational complexity of framing the acoustic data into short overlapping frames is not considered since it is merely a buffering operation.

The computational complexity of the sound activity detector is expressed by N multiply-accumulates and one multiplication to compute the average energy according to Equation 5.1. The average energy is then evaluated by a comparator to determine whether the energy level exceeds the predefined threshold or not. Saving the threshold (scalar) and the SAD detected frame requires a memory allocation of $1 \cdot S$ bytes and $N \cdot S$ bytes respectively.

The windowing operation results in N multiplications per SAD detected frame and requires a memory allocation of $N \cdot S$ bytes for saving the Hamming window parameters and $N \cdot S$ bytes for storing the windowed framed.

Transforming the windowed frame to the frequency domain using the Cooley-Tukey fast Fourier transformation (FFT) algorithm results in $K \cdot \log_2(K)$ complex multiply-accumulates, with $K = 2^{\lceil \log_2(N) \rceil}$ being the number of frequency bins which is set to the next radix-2 number of N, and $1/2 \cdot K$ absolute value operations to retain the lowest half of the magnitude spectrum. Saving the FFT parameters (i.e. complex-valued butterfly diagram weights) and the computed magnitude spectrum requires a memory allocation of $K \cdot S$ bytes and $(1/2 \cdot K) \cdot S$ bytes respectively [36].

Mapping the magnitude spectrum to the MEL-scale basically involves a matrix multiplication with a MEL-filter bank. This operation involves $1/2 \cdot K \cdot B$ multiply-accumulates, with B denoting the number of bands in the MEL-filter bank, and requires $(1/2 \cdot K \cdot B) \cdot S$ bytes of memory for storing the filter weights. However, it is worth to note that previous two expressions are in fact an overestimation since many MEL-filter weights are zeros and thus can be omitted during the calculations. Computing the log-transformed MEL-magnitude features involves B natural logarithms and requires a memory allocation of $B \cdot S$ bytes of storage.

The MFCCs are obtained by applying a discrete cosine transformation (DCT) on the log-transformed MEL-features. This operation requires $B \cdot D$ multiply-accumulates, with D as the total number of cepstral coefficients, and a memory allocation of $(B \cdot D) \cdot S$ bytes for storing the DCT-matrix. In total F feature feature frames must be stored to compute the cepstral statistics resulting in an overall memory allocation of $(F \cdot D) \cdot S$ bytes

The final step is the computation of the cepstral means and standard deviations over a window of F frames. This operation results in $3 \cdot D \cdot F$ additions, $(2 + F) \cdot D$ multiplications and D square roots. Saving the $\mu\sigma$-MFCC feature vector requires a memory allocation of $(2 \cdot D) \cdot S$ bytes.

As already discussed in Section 3.4 of Chapter 3, the most dominant parameter in controlling the computational complexity of the acoustic pre-processing stage is the sampling frequency. The sampling frequency defines the number of samples per frame which on its turns determines the number of frequency bins. The number of frequency bins defines the computational complexity of the FFT operation and is also linked to the computational complexity of the MEL-frequency warping stage. The most important parameter in controlling the required memory allocation is the used data type precision. Lowering the precision reduces the required memory allocation for saving the feature extraction parameters and the processed data. Furthermore, it is also worth

to note that the data precision is also linked to the overall computational complexity since lower precisions yield faster computations.

The used feature configuration parameters during the experiments of this chapter are: $N = 800$ given a sampling frequency of 32 kHz and a frame length of 25 ms, $K = 1024$ as the next radix-2 number of N, $B = 40$ and $D = 14$. The $\mu\sigma$-MFCC feature vectors are computed over 500 ms long windows, i.e. $F = 50$, and with an overlap of 250 ms. As a result, a 28th-dimensional $\mu\sigma$-MFCC feature vector is computed every 250 ms. All computations are done using double-precision floating point representation (64 bits).

5.2.2.3 Local classifier

The aim of the local classifier is to analyse the extracted $\mu\sigma$-MFCC features for the purpose of acoustic presence detection. The focus of the classifier is situated in discriminating '*presence*' from '*absence*' related sound data, i.e. a binary classification task, and thus not in classifying individual sound events. The work proposed in this chapter will mainly concentrate on the development of an '*adaptive*' and '*resource efficient*' learning algorithm which can be embedded locally on devices with limited computational resources (e.g. ARM Cortex mobile M-series). In this work, a '*least-squares support vector machine*' (LS-SVM) framework is adapted for the purpose of acoustic presence detection. LS-SVM involves solving a set of linear equations instead of a convex quadratic programming (QP) problem as with classical SVMs. Such processing can be easily mapped to an embedded platform using already available optimised microcontroller unit libraries. The standard LS-SVM formulation is known to not result in a sparse solution and thus become computationally expensive to solve in case of large-scale datasets [82]. In this work, two equivalent approximation schemes will be investigated to obtain a sparse solution and are '*fixed-size*' LS-SVM [45] and '*subset-of-regressors*' LS-SVM [79, 150]. Both approximations result in a fixed number of support vectors or prototypes that can be used to model the feature space. Being able to control the number of support vectors gives direct access in controlling the computational complexity of the classifier. The latter is a very interesting aspect when the algorithm needs to be mapped to an embedded device with limited resources. Furthermore, linear systems are also easy to modify such that the model parameters can be learned in an adaptive fashion using recursive least-squares (RLS) [79, 115]. All these properties makes LS-SVM based learning very suitable for the purpose of on-edge and adaptive presence detection.

5.3 On-edge presence detection classifiers

This section introduces a LS-SVM based on-edge acoustic presence detection classifier. First, the general form of a SVM is restated before introducing the LS-SVM methodology. The main differences between both will be highlighted together with their advantages and disadvantages. Next, two approximation schemes are introduced to obtain a sparse solution in case of large-scale datasets which are fixed-size LS-SVM and subset-of-regressors LS-SVM.

During the remainder of this chapter the following notation will be maintained unless stated otherwise. The dataset is given by $\{(\mathbf{x}_i, y_i)\}_{i=1}^{t}$ where each $\mathbf{x}_i \in \mathbb{R}^D$ is a column vector in the D-dimensional input space, i.e the $\mu\sigma$-MFCC features, and $y_i \in \{-1, +1\}$ its corresponding class label, i.e '*absence*' and '*presence*' respectively. The subscript i denotes the time index when the feature vector and its corresponding class label is recorded.

5.3.1 Support vector machine

As already stated in Section 2.3.2, the objective of a SVM is to find a decision function that separates the set of data points of a binary classification problem such that the margin, i.e. distance of the closest points to the hyperplane, is maximised. The general form of a SVM is given by

$$\min_{\mathbf{w}\in\mathbb{R}^{D_\varphi}} J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{i=1}^{t}\xi_i, \tag{5.2}$$

subject to (s.t.) $y_i(\mathbf{w}^\top\varphi(\mathbf{x}_i)) \geq 1-\xi_i$ and $\xi_i \geq 0$, $\forall i = [1, 2, \ldots, t]$, and is known as the soft-margin primal formulation. Notice that the intercept b is neglected for notational simplicity. The feature map $\varphi : \mathcal{X} \to \mathcal{V}$ transforms the non-linear separable input data $\mathcal{X} \in \mathbb{R}^D$ to a higher dimensional feature space $\mathcal{V} \in \mathbb{R}^{D_\varphi}$ in which linear classification can be applied. The trade-off between margin maximisation and the allowable number misclassifications is controlled by the regularisation parameter C. Classifying an unseen sample $\bar{\mathbf{x}}$ in the primal space is done by

$$\hat{y} = \text{sign}(\mathbf{w}^\top\varphi(\bar{\mathbf{x}})). \tag{5.3}$$

Rewriting 5.2 using the method of Lagrange multipliers yield the following dual formulation

$$\alpha_1^*, \ldots, \alpha_t^* = \max_{\alpha_1,\ldots,\alpha_t} -\frac{1}{2}\sum_{i=1}^{t}\sum_{j=1}^{t}\alpha_i\alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i=1}^{t}\alpha_i, \tag{5.4}$$

s.t. $\sum_{i=1}^{t} \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$, $i = [1, 2, \ldots, t]$. The kernel trick is defined by $k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^\top \varphi(\mathbf{x}_j)$ and can be seen as a similarity measure between $\mathbf{x}_i$ and $\mathbf{x}_j$. Equation 5.4 shows that searching for the maximum-margin decision boundary is equivalent to searching for the set of support vectors, e.g. the training examples with non-zero Lagrange multipliers α_i. Classifying an unseen sample $\bar{\mathbf{x}}$ is now done by

$$\hat{y} = \text{sign}\big[\sum_{i=1}^{t} y_i \alpha_i k(\bar{\mathbf{x}}, \mathbf{x}_i)\big], \tag{5.5}$$

with $\hat{y}$ as the predicted class label.

Although SVMs are proven to be a very powerful tool in modelling complex feature spaces, the default formulation lacks the ability to control the number of support vectors. Research has shown that the number of support vectors is strongly related to the size and complexity of the data set, and that SVMs typically tend to use a large number of support vectors to model the feature space [19, 126]. Having no direct control over the support vectors impedes a practical implementation since it defines the overall computational complexity of the model, i.e. processing power and memory allocation, and thereby defining the required embedded platform specifications.

5.3.2 Least-squares SVM

The least-squares support vector machine (LS-SVM) approach preserves the SVM methodology but introduces a simplification via equality constraints and a least-squares optimisation [161]. Hence, the primal formulation of LS-SVM is defined by

$$\min_{\mathbf{w} \in \mathbb{R}^{D_\varphi}} J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w} + \frac{\gamma}{2}\sum_{i=1}^{t} \epsilon_i^2, \tag{5.6}$$

s.t. $\mathbf{w}^\top \varphi(\mathbf{x}_i) = y_i - \epsilon_i$, $\forall i = [1, 2, \ldots, t]$, with $\varphi : \mathbb{R}^D \to \mathbb{R}^{D_\varphi}$ denoting the feature map, γ the regularisation parameter and $\epsilon_i = y_i - \hat{y}_i$ the prediction error. The intercept b is again neglected for notational simplicity. Classifying an unseen sample $\bar{\mathbf{x}}$ in the primal formulation of LS-SVM is done similarly as with the primal formulation of SVM by using Equation 5.3.

Rewriting Equation 5.6 using the method of Lagrange multipliers yields

$$\mathcal{L}(\mathbf{w}, \epsilon_i; \alpha_i) = \frac{1}{2}\mathbf{w}^\top \mathbf{w} + \frac{\gamma}{2}\sum_{i=1}^{t} \epsilon_i^2 - \sum_{i=1}^{t} \alpha_i (\mathbf{w}^\top \varphi(\mathbf{x}_i) - y_i + \epsilon_i), \tag{5.7}$$

with $\alpha_i \in \mathbb{R}$ as the Lagrange multipliers. The Lagrangian can be minimised by computing the partial derivatives $\frac{\partial \mathcal{L}}{\partial \mathbf{w}}$, $\frac{\partial \mathcal{L}}{\partial \epsilon_i}$ and $\frac{\partial \mathcal{L}}{\partial \alpha_i}$, $\forall i = [1, 2, \ldots, t]$, and setting them to zero. Rewriting the solution in terms of a matrix notation results in following optimality condition

$$\boldsymbol{\alpha} = (\boldsymbol{\Omega} + \tfrac{1}{\gamma}\mathbf{I})^{-1}\mathbf{y}, \tag{5.8}$$

with $\boldsymbol{\Omega} \in \mathbb{R}^{t \times t}$ being the dense kernel matrix, i.e. $[\boldsymbol{\Omega}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{I}$ as the identity matrix, and $\mathbf{y} \in \mathbb{R}^t$ the column-wise label vector. As one can see, solving the dual form of LS-SVM given by Equation 5.8 is basically solving a set of linear equations. Predicting an unseen sample $\bar{\mathbf{x}}$ in the dual form of LS-SVM is done similarly as with the dual formulation of SVM by using Equation 5.5.

The main disadvantage of LS-SVM is that it does not result in a sparse solution, i.e. each data point turns into a support vector with a non-zero weight α_i, and thereby becoming computationally expensive to solve in case of large datasets $\mathcal{O}(t^3)$. For large-scale applications it is thus often advantageous to solve the problem in its primal form where the dimension of the parameter vector $\mathbf{w} \in \mathbb{R}^D$ is much smaller compared to $\boldsymbol{\alpha} \in \mathbb{R}^t$, or by approximating the dense kernel matrix in the dual form on arbitrary points through linear combination of kernels sampled from the data. Solving the primal results into the well-known fixed-size least-squares support vector machine (FS-LSSVM) solution [161]. Approximating the dense kernel matrix is known in the literature by the '*subset of regressors*' approach [79, 150] which we will further abbreviate by subset-of-regressors least-squares support vector machine (SR-LSSVM).

5.3.3 Fixed-size least-squares SVM

The primal formulation of LS-SVM is given by Equation 5.6 and can be rewritten in

$$\min_{\mathbf{w} \in \mathbb{R}^{D_\varphi}} J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{t} \left(y_i - \mathbf{w}^\top \varphi(\mathbf{x}_i)\right)^2 + \frac{\gamma}{2} \mathbf{w}^\top \mathbf{w}, \tag{5.9}$$

which is known as the standard formulation of a linear ridge regression model [60]. In order to work in the primal space using a kernel function other than the linear one, it is required to compute an explicit approximation of the feature map $\hat{\varphi} : \mathbb{R}^D \to \mathbb{R}^m$ with $m \ll t$, such that $[\boldsymbol{\Omega}]_{ij} \approx \hat{\varphi}(\mathbf{x}_i)^\top \hat{\varphi}(\mathbf{x}_j)$. This leads to a sparse representation of the model when estimating in the primal space [45].

Explicit expressions for $\hat{\varphi}(\cdot)$ can be obtained from m prototype vectors[1] using the Nyström approximation [181]. Skipping the mathematical derivations, the

[1]Note: the prototype vector selection is discussed in Section 5.5.4

explicit approximation of the features in matrix notation is given by

$$\boldsymbol{\Phi}_{(tm)} = \boldsymbol{\Omega}_{(tm)} \mathbf{U} \boldsymbol{\Lambda}^{-\frac{1}{2}}, \tag{5.10}$$

with matrix $\boldsymbol{\Phi}_{(tm)} \in \mathbb{R}^{t \times m}$ defined as

$$\boldsymbol{\Phi}_{(tm)} = \begin{bmatrix} \hat{\varphi}_1(\mathbf{x}_1) & \hat{\varphi}_2(\mathbf{x}_1) & \dots & \hat{\varphi}_m(\mathbf{x}_1) \\ \hat{\varphi}_1(\mathbf{x}_2) & \hat{\varphi}_2(\mathbf{x}_2) & \dots & \hat{\varphi}_m(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\varphi}_1(\mathbf{x}_t) & \hat{\varphi}_2(\mathbf{x}_t) & \dots & \hat{\varphi}_m(\mathbf{x}_t) \end{bmatrix}, \tag{5.11}$$

and $\boldsymbol{\Omega}_{(tm)} \in \mathbb{R}^{t \times m}$, $[\boldsymbol{\Omega}_{(tm)}]_{ij} = k(\mathbf{x}_i, \tilde{\mathbf{x}}_j)$, $\mathbf{x}_i \in \mathbf{X}_{tr}$ the training set, and $\tilde{\mathbf{x}}_j \in \mathbf{X}_{pv}$ the set of prototypes. The square matrix of eigenvectors $\mathbf{U}$ and the diagonal matrix of eigenvalues $\boldsymbol{\Lambda}$ are obtained by an eigenvalue decomposition on the kernel matrix $\boldsymbol{\Omega}_{(mm)} \in \mathbb{R}^{m \times m}$ with $[\boldsymbol{\Omega}_{(mm)}]_{ij} = k(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$.

Rewriting Equation 5.9 in terms of a matrix notation with the approximated features $\boldsymbol{\Phi}_{(tm)}$ yields

$$\min_{\mathbf{w} \in \mathbb{R}^m} J(\mathbf{w}) = \frac{1}{2} \|\mathbf{y} - \boldsymbol{\Phi}_{(tm)} \mathbf{w}\|^2 + \frac{\gamma}{2} \mathbf{w}^\top \mathbf{w}, \tag{5.12}$$

and is solved by

$$\mathbf{w} = (\boldsymbol{\Phi}_{(tm)}^\top \boldsymbol{\Phi}_{(tm)} + \tfrac{1}{\gamma} \mathbf{I})^{-1} \boldsymbol{\Phi}_{(tm)}^\top \mathbf{y}. \tag{5.13}$$

Classifying an unseen sample $\bar{\mathbf{x}}$ in the primal space is now done by

$$\hat{y} = \operatorname{sign}(\mathbf{w}^\top \hat{\varphi}(\bar{\mathbf{x}})). \tag{5.14}$$

The approximated feature map $\hat{\varphi}(\cdot)$ results in a sparse solution and thereby reducing the overall computational complexity from $\mathcal{O}(t^3)$ to $\mathcal{O}(mt^2)$. By using a recursive least-squares (RLS) implementation, the parameter vector $\mathbf{w}$ can be efficiently updated from the newly collected samples without the need of a complete retraining phase [41]. However, the main disadvantage of FS-LSSVM is that the feature map is approximated by the eigenvectors and eigenvalues of the kernel matrix $\boldsymbol{\Omega}_{(mm)}$. This impedes an adaptive solution where the prototypes may vary over time. More specifically, adding or removing prototypes from $\mathbf{X}_{pv}$ changes $\mathbf{U}$ and $\boldsymbol{\Sigma}$, i.e. the eigenvectors and eigenvalues of $\boldsymbol{\Omega}_{(mm)}$, and thereby resulting in a different feature map $\hat{\varphi}(\cdot)$. As a result, the learned parameter vector $\mathbf{w}$ does not correspond any more to the new feature map and cannot be updated without recalling all past seen samples. A solution to overcome this problem is to use an alternative representation, i.e. the subset-of-regressors approach, which facilitates a complete adaptive learning strategy where both the parameter vector and the set of prototypes can be updated given the newly collected data sample only.

5.3.4 Subset-of-regressors least-squares SVM

The subset-of-regressors LS-SVM (SR-LSSVM) approximates the dense kernel matrix $\mathbf{\Omega}$ on arbitrary points through linear combination of kernels sampled from the data instead of searching for an approximated feature map $\hat{\varphi}(\cdot)$. Before explaining the SR-LSSVM methodology, let us first introduce the following equivalent notation. According to Equation 5.10, the mapped features are defined by

$$\mathbf{\Phi}_{(tm)} = \mathbf{\Omega}_{(tm)}\mathbf{U}\mathbf{\Lambda}^{-\frac{1}{2}}, \tag{5.15}$$

which is equivalent to

$$\mathbf{\Phi}_{(tm)} = \mathbf{\Omega}_{(tm)}\mathbf{\Omega}_{(mm)}^{-\frac{1}{2}}, \tag{5.16}$$

since it can be shown that $\mathbf{U}\mathbf{\Lambda}^{-\frac{1}{2}} = \mathbf{\Omega}_{(mm)}^{-\frac{1}{2}}$ (See appendix 7.3 for the derivation). Plugging this equivalent formulation of $\mathbf{\Phi}_{(tm)}$ into Equation 5.12, the primal form of FS-LSSVM becomes

$$\min_{\mathbf{w}\in\mathbb{R}^m} J(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - \mathbf{\Omega}_{(tm)}\mathbf{\Omega}_{(mm)}^{-\frac{1}{2}}\mathbf{w}\|^2 + \frac{\gamma}{2}\mathbf{w}^\top\mathbf{w}, \tag{5.17}$$

which is the same as solving

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^m} J(\boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{y} - \mathbf{\Omega}_{(tm)}\boldsymbol{\beta}\|^2 + \frac{\gamma}{2}\boldsymbol{\beta}^\top\mathbf{\Omega}_{(mm)}^{-1}\boldsymbol{\beta}, \tag{5.18}$$

with $\boldsymbol{\beta} = \mathbf{\Omega}_{(mm)}^{-\frac{1}{2}}\mathbf{w}$. The optimal solution is now given by

$$\boldsymbol{\beta} = (\mathbf{\Omega}_{(tm)}^\top\mathbf{\Omega}_{(tm)} + \tfrac{1}{\gamma}\mathbf{\Omega}_{(mm)})^{-1}\mathbf{\Omega}_{(tm)}^\top\mathbf{y}, \tag{5.19}$$

and classifying an unseen test sample is done according to

$$\hat{y} = \text{sign}(\boldsymbol{\beta}^\top\hat{\varphi}'(\bar{\mathbf{x}})), \tag{5.20}$$

with $\hat{\varphi}'(\bar{\mathbf{x}}) = [k(\bar{\mathbf{x}}, \tilde{\mathbf{x}}_1), k(\bar{\mathbf{x}}, \tilde{\mathbf{x}}_2), \ldots, k(\bar{\mathbf{x}}, \tilde{\mathbf{x}}_m)]$ being the column-wise kernel vector between the test sample $\bar{\mathbf{x}}$ and the set of prototypes $[\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_m] \in \mathbf{X}_{pv}$.

Although the notation of Equation 5.19 still requires all past seen examples to update $\boldsymbol{\beta}$ when prototypes are added or removed from $\mathbf{X}_{pv}$, it paves the way for a complete adaptive learning strategy using the subset-of-regressors approach as proposed in [41, 79]. The latter is done by approximating the kernel on points through linear combination of kernels from the prototypes, i.e. $\hat{\varphi}'(\mathbf{x}) \approx \sum_{i=1}^m a_i\hat{\varphi}'(\tilde{\mathbf{x}}_i)$. The weights a_i are determined such that

$$\delta = \min_{\mathbf{a}} \|\hat{\varphi}'(\mathbf{x}) - \sum_{i=1}^m a_i\hat{\varphi}'(\tilde{\mathbf{x}}_i)\|^2, \tag{5.21}$$

is minimised for a given $\mathbf{x}$ with $\mathbf{a} = [a_1, a_2, \ldots, a_m]$. The solution of this problem is given by

$$\mathbf{a} = \boldsymbol{\Omega}_{(mm)}^{-1} \hat{\varphi}^{'}(\mathbf{x}), \tag{5.22}$$

By defining the $t \times m$ weighting matrix $\mathbf{A}_{(tm)}$ having on its rows $\mathbf{a}_i = (\boldsymbol{\Omega}_{(mm)}^{-1} \hat{\varphi}^{'}(\mathbf{x}_i))^\top$, $i = [1, \ldots, t]$, Equation 5.22 becomes

$$\boldsymbol{\Omega}_{(tm)} \approx \tilde{\boldsymbol{\Omega}}_{(tm)} = \mathbf{A}\boldsymbol{\Omega}_{(mm)}, \tag{5.23}$$

where $\tilde{\boldsymbol{\Omega}}_{tm}$ is the approximated version of $\boldsymbol{\Omega}_{tm}$. Hence, SR-LSSVM is defined by plugging $\tilde{\boldsymbol{\Omega}}_{tm}$ into Equation 5.18 such that optimality is obtained by solving

$$\boldsymbol{\beta} = (\tilde{\boldsymbol{\Omega}}_{(tm)}^\top \tilde{\boldsymbol{\Omega}}_{(tm)} + \tfrac{1}{\gamma}\boldsymbol{\Omega}_{(mm)})^{-1} \tilde{\boldsymbol{\Omega}}_{(tm)}^\top \mathbf{y}. \tag{5.24}$$

The crucial advantage of this representation is that both $\mathbf{X}_{pv}$ and $\boldsymbol{\beta}$ can be updated using a recursive least-squares implementation without the need of recalling all past seen examples. A detailed explanation of the RLS implementation is given in the next section.

5.4 Adaptive learning

As already discussed in the previous section, SR-LSSVM enables to update both the set of prototypes $\mathbf{X}_{pv}$ and the parameter vector $\boldsymbol{\beta}$ each time a new sample pair $\{\mathbf{x}_i, y_i\}$ is presented to the algorithm. Let us now define t as the current time index, $(\mathbf{x}_t, y_t)$ the currently observed data sample and class label, $\{(\mathbf{x}_i, y_i)\}_{i=1}^{t-1}$ the past $t-1$ samples and their corresponding class labels, $\{\tilde{\mathbf{x}}_m\}_{j=1}^{m}$ the set of m selected prototypes, and $\mathbf{P}_{(tm)}$ as the $m \times m$ cross-product matrix given by $\mathbf{P}_{(tm)} = \tilde{\boldsymbol{\Omega}}_{(tm)}^\top \tilde{\boldsymbol{\Omega}}_{(tm)} + \frac{1}{\gamma}\boldsymbol{\Omega}_{(mm)}$. Note that a double index is used to indicate both the dependence on the number of samples t and the number of prototypes m. The solution of Equation 5.19 can now be expressed by $\boldsymbol{\beta}_{(tm)} = \mathbf{P}_{(tm)}^{-1} \tilde{\boldsymbol{\Omega}}_{(tm)}^\top \mathbf{y}_{(t)}$ with an incurred cost of $\zeta_{(tm)} = J(\boldsymbol{\beta}_{(tm)})$. Each time a new sample is presented to the algorithm, one or more of the following steps are performed as proposed in [79]:

- **Normal step**: from $\{\mathbf{P}_{(t-1,m)}^{-1}, \boldsymbol{\beta}_{(t-1,m)}, \zeta_{(t-1,m)}\}$ to $\{\mathbf{P}_{(tm)}^{-1}, \boldsymbol{\beta}_{(tm)}, \zeta_{(tm)}\}$
  Updates the inverse cross-product matrix and the parameter vector given the currently observed sample $\mathbf{x}_t$ and its corresponding class label y_t. The set of prototypes $\mathbf{X}_{pv}$ are not altered in this step.

- **Growing step:** from $\{\mathbf{P}^{-1}_{(tm)}, \boldsymbol{\beta}_{(tm)}, \zeta_{(tm)}\}$ to $\{\mathbf{P}^{-1}_{(t,m+1)}, \boldsymbol{\beta}_{(t,m+1)}, \zeta_{(t,m+1)}\}$
  If the current sample is sufficiently different from the prototypes in $\mathbf{X}_{pv}$ or it strongly contributes to the solution of the problem, then $\mathbf{x}_t$ is added to $\mathbf{X}_{pv}$ and becomes a prototype vector. Hence, the number of prototype vectors in the model is increased by 1.
- **Pruning step:** from $\{\mathbf{P}^{-1}_{(tm)}, \boldsymbol{\beta}_{(tm)}, \zeta_{(tm)}\}$ to $\{\mathbf{P}^{-1}_{(t,m\setminus i)}, \boldsymbol{\beta}_{(t,m\setminus i)}, \zeta_{(t,m\setminus i)}\}$
  If the maximum number of prototypes is exceeded, the algorithm will remove the prototype in $\mathbf{X}_{pv}$ that contributes the least to the solution.

Before introducing the update formulas of each step, let us first recall two matrix identities for recursively computing the inverse of a matrix [66]:

$$\text{(1) If } \mathbf{B}_{(t+1)} = \mathbf{B}_{(t)} + \mathbf{b}\mathbf{b}^\top \text{ then } \mathbf{B}^{-1}_{(t+1)} = \mathbf{B}^{-1}_{(t)} - \frac{\mathbf{B}^{-1}_{(t)}\mathbf{b}\mathbf{b}^\top\mathbf{B}^{-1}_{(t)}}{1+\mathbf{b}^\top\mathbf{B}^{-1}_{(t)}\mathbf{b}}, \tag{5.25}$$

$$\text{(2) If } \mathbf{B}_{(t+1)} = \begin{bmatrix}\mathbf{B}_{(t)} & \mathbf{b}\\ \mathbf{b}^\top & b^*\end{bmatrix} \text{ then}$$

$$\mathbf{B}^{-1}_{(t+1)} = \begin{bmatrix}\mathbf{B}^{-1}_{(t)} & \mathbf{0}\\ \mathbf{0}^\top & 0\end{bmatrix} + \frac{1}{\Delta_u}\begin{bmatrix}-\mathbf{u}\\ 1\end{bmatrix}\begin{bmatrix}-\mathbf{u}\\ 1\end{bmatrix}^\top, \tag{5.26}$$

with $\Delta_u = b^* - \mathbf{b}^\top\mathbf{u}$ and $\mathbf{u} = \mathbf{B}^{-1}_{(t)}\mathbf{b}$.

5.4.1 Normal step

Assuming that $\{\mathbf{P}^{-1}_{(t-1,m)}, \boldsymbol{\beta}_{(t-1,m)}, \zeta_{(t-1,m)}\}$ are known from previous computations, the objective is to update the model to $\{\mathbf{P}^{-1}_{(tm)}, \boldsymbol{\beta}_{(tm)}, \zeta_{(tm)}\}$ every time a new sample is presented to the algorithm. Let us define the column-wise kernel vector between the currently observed sample and the set of prototypes by $\hat{\varphi}'(\mathbf{x}_t) = [k(\mathbf{x}_t, \tilde{\mathbf{x}}_1), k(\mathbf{x}_t, \tilde{\mathbf{x}}_2), \ldots, k(\mathbf{x}_t, \tilde{\mathbf{x}}_m)]$, then the design matrix and label vector are given by

$$\tilde{\boldsymbol{\Omega}}_{(tm)} = \begin{bmatrix}\tilde{\boldsymbol{\Omega}}_{(t-1,m)}\\ \hat{\varphi}'(\mathbf{x}_t)^\top\end{bmatrix} \quad \text{and} \quad \mathbf{y}_{(t)} = \begin{bmatrix}\mathbf{y}_{(t-1)}\\ y_t\end{bmatrix}. \tag{5.27}$$

The cross-product matrix is updated by $\mathbf{P}_{(tm)} = \mathbf{P}_{(t-1,m)} + \hat{\varphi}'(\mathbf{x}_t)\hat{\varphi}'(\mathbf{x}_t)^\top$ which results in following set of well-known RLS updates:

$$\mathbf{P}_{(tm)}^{-1} = \mathbf{P}_{(t-1,m)}^{-1} - \frac{\mathbf{P}_{(t-1,m)}^{-1}\hat{\varphi}'(\mathbf{x}_t)\hat{\varphi}'(\mathbf{x}_t)^\top\mathbf{P}_{(t-1,m)}^{-1}}{\Delta}, \tag{5.28}$$

$$\boldsymbol{\beta}_{(tm)} = \boldsymbol{\beta}_{(t-1,m)} + \frac{\epsilon}{\Delta}\mathbf{P}_{(t-1,m)}^{-1}\hat{\varphi}'(\mathbf{x}_t), \tag{5.29}$$

$$\zeta_{(tm)} = \zeta_{(t-1,m)} + \frac{\epsilon^2}{\Delta}, \tag{5.30}$$

with $\Delta = 1 + \hat{\varphi}'(\mathbf{x}_t)^\top\mathbf{P}_{(t-1,m)}^{-1}\hat{\varphi}'(\mathbf{x}_t)$ and $\epsilon = y_t - \boldsymbol{\beta}_{(t-1,m)}^\top\hat{\varphi}'(\mathbf{x}_t)$. The overall computational complexity for these updates is $\mathcal{O}(m^2)$. Note that these RLS updates can also be applied to FS-LSSVM since the set of prototypes in $\mathbf{X}_{pv}$ is not altered. See Appendix 7.4 for the updates of FS-LSSVM.

5.4.2 Growing step

The growing step adds an additional prototype vector to the set of prototypes such that $\tilde{\mathbf{x}}_{m+1}$ is defined by $\mathbf{x}_t$. Skipping the mathematical derivations given in [79], the following set of updates are obtained:

$$\mathbf{P}_{(t,m+1)}^{-1} = \begin{bmatrix}\mathbf{P}_{(tm)}^{-1} & \mathbf{0}\\ \mathbf{0}^\top & 0\end{bmatrix} + \frac{1}{\Delta_u}\begin{bmatrix}-\mathbf{u}\\ 1\end{bmatrix}\begin{bmatrix}-\mathbf{u}\\ 1\end{bmatrix}^\top, \tag{5.31}$$

$$\boldsymbol{\beta}_{(t,m+1)} = \begin{bmatrix}\boldsymbol{\beta}_{(tm)}\\ 0\end{bmatrix} + \kappa\begin{bmatrix}\mathbf{u}\\ 1\end{bmatrix}, \tag{5.32}$$

$$\zeta_{(t,m+1)} = \zeta_{(tm)} - \kappa^2\Delta_u, \tag{5.33}$$

with

$$\mathbf{u} = \mathbf{a} + \frac{\delta}{\Delta}\mathbf{P}_{(t-1,m)}^{-1}\hat{\varphi}'(\tilde{\mathbf{x}}_{m+1}), \tag{5.34}$$

$$\Delta_u = \frac{\delta^2}{\Delta} + \gamma\delta, \tag{5.35}$$

$$\kappa = -\frac{\delta\epsilon}{\Delta\Delta_u}, \tag{5.36}$$

where $\delta = \phi^* - \hat{\varphi}'(\tilde{\mathbf{x}}_{m+1})^\top\mathbf{a}$, with $\phi^* = k(\tilde{\mathbf{x}}_{(m+1)}, \tilde{\mathbf{x}}_{(m+1)})$, and $\mathbf{a} = \boldsymbol{\Omega}_{(mm)}^{-1}\hat{\varphi}'(\tilde{\mathbf{x}}_{m+1})$ according to Equations 5.21 and 5.22 respectively.

Finally, every time a prototype is added to $\mathbf{X}_{pv}$ the kernel matrix $\mathbf{\Omega}_{(mm)}$ and its inverse must be updated according to

$$\mathbf{\Omega}_{(m+1,m+1)} = \begin{bmatrix} \mathbf{\Omega}_{(mm)} & \hat{\varphi}'(\tilde{\mathbf{x}}_{m+1}) \\ \hat{\varphi}'(\tilde{\mathbf{x}}_{m+1})^\top & \phi^* \end{bmatrix}, \tag{5.37}$$

and

$$\mathbf{\Omega}^{-1}_{(m+1,m+1)} = \begin{bmatrix} \mathbf{\Omega}^{-1}_{(mm)} & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix} + \frac{1}{\delta} \begin{bmatrix} -\mathbf{a} \\ 1 \end{bmatrix} \begin{bmatrix} -\mathbf{a} \\ 1 \end{bmatrix}^\top, \tag{5.38}$$

respectively. The overall computational complexity of adding a prototype is $\mathcal{O}(m^2)$.

Deciding whether or not the current sample $\mathbf{x}_t$ must be included to the set of prototypes is done by a two-part criterion as proposed in [79]. The first part measures the '*novelty*' of the current sample, i.e. whether it is sufficiently different from those already stored in $\mathbf{X}_{pv}$, and is expressed by δ (restated from Equation 5.21). The second part determines the '*usefulness*' of the candidate prototype, i.e. the reduction in regularised cost, and is given by $\kappa^2 \cdot \Delta_u$ according to Equation 5.33. The authors in [79] combine both criteria such that only the samples having $(\delta \cdot \kappa^2 \cdot \Delta_u) > \tau$ will be added to $\mathbf{X}_{pv}$.

5.4.3 Pruning step

Once the maximum number of prototypes is reached, the pruning step removes the prototype that contributes the least to the reduction in cost. First consider the case when trying to delete the last prototype in $\mathbf{X}_{pv}$. Again skipping the mathematical derivations given in [79], and by switching the role of old and new in Equations 5.31, 5.32, and 5.33 respectively, the following set of updates are obtained:

$$\begin{bmatrix} \mathbf{\Omega}^{-1}_{(m-1,m-1)} & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix} = \mathbf{\Omega}^{-1}_{(mm)} - \frac{1}{v} \begin{bmatrix} \mathbf{v} \\ -1 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ -1 \end{bmatrix}^\top, \tag{5.39}$$

$$\begin{bmatrix} \mathbf{P}^{-1}_{(t,m-1)} & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix} = \mathbf{P}^{-1}_{(tm)} - \frac{1}{u} \begin{bmatrix} \mathbf{u} \\ -1 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ -1 \end{bmatrix}^\top, \tag{5.40}$$

$$\begin{bmatrix} \boldsymbol{\beta}_{(t,m-1)} \\ 0 \end{bmatrix} = \boldsymbol{\beta}_{(tm)} + \frac{b}{u} \begin{bmatrix} \mathbf{u} \\ -1 \end{bmatrix}, \tag{5.41}$$

$$\zeta_{(t,m-1)} = \zeta_{(tm)} + \frac{b^2}{u}, \tag{5.42}$$

with

$$\mathbf{v} = \mathbf{\Omega}^{-1}_{(mm)}[1:m-1,m] \quad \text{and} \quad v = \mathbf{\Omega}^{-1}_{(mm)}[m,m], \tag{5.43}$$

$$\mathbf{u} = \mathbf{P}^{-1}_{(tm)}[1:m-1,m] \quad \text{and} \quad u = \mathbf{P}^{-1}_{(tm)}[m,m], \tag{5.44}$$

$$b = \boldsymbol{\beta}_{(tm)}[m], \tag{5.45}$$

with the elements between the square brackets denoting the row and column indices respectively.

Deleting a prototype other then the last one is done by reordering the indices of the prototypes in $\mathbf{X}_{pv}$ and swapping the ith row/column in $\boldsymbol{P}^{-1}_{(tm)}$, $\boldsymbol{\beta}_{(tm)}$, $\mathbf{\Omega}^{-1}_{(mm)}$ and $\mathbf{\Omega}_{(mm)}$ before applying the corresponding pruning updates. Obtaining $\mathbf{\Omega}_{(m-1,m-1,)}$ is simply done by removing the last row/column of $\mathbf{\Omega}_{(mm)}$. Identifying the prototype in $\mathbf{X}_{pv}$ having the least impact to the cost function is done by

$$i = \underset{i}{\operatorname{argmin}} \frac{\boldsymbol{\beta}_{tm}(i)^2}{\mathbf{P}^{-1}_{tm}(i,i)}, \quad \text{with} \quad i = 1,\ldots,m. \tag{5.46}$$

The overall complexity of deleting a prototype is again $\mathcal{O}(m^2)$.

5.5 Experimental specific details

5.5.1 Recorded dataset

A real-life acoustic dataset was recorded in an office environment for the development and evaluation of the proposed adaptive learning strategies. The outer dimensions of the office environment are 7.2 by 6.6 meters giving space for six medium sized desks. A detailed floor plan of the used office environment together with the sensor locations is shown in Figure 5.2. As one can see, the acoustic sensor was placed next to the door entrance at an height of approximately 1.2 meters. This location was motivated by installation considerations, i.e. nearby an electrical power supply, and with the assumption that people can only enter or leave the office by passing the microphone.

In total 555 hours of acoustic data, i.e. 105 hours of presence and 450 hours of absence, was recorded over a period of 28 days. A more detailed overview of the collected data is given by Table 5.2. The complete acquisition architecture was controlled by a Raspberry Pi in combination with a graphical user interface (GUI). The GUI allowed users to start and stop the recordings if needed and to label the data on-the-fly by means of a simple button press when entering or

leaving the room. In addition, magnetic reed switches were placed at the door and windows and were connected to the Raspberry Pi to keep track of their status, i.e. whether they opened or closed.

Table 5.2: Overview of the recorded dataset.

	Number of persons presence					
	0	1	2	3	4	5
Acoustic data [in h]	450	105				
		61.2	30.5	9.8	1.4	2.1
Door status [in %]	closed: 94.0% - open: 6.0%					
Window A status [in %]	closed: 68.7% - open: 31.3%					
Window B status [in %]	closed: 10.0% - open: 90.0%					

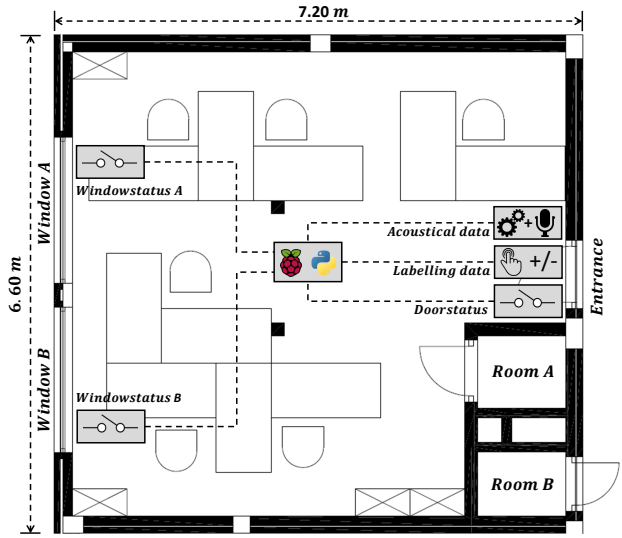

Figure 5.2: Floor plan of the used office environment together with the desk setup and sensor locations. The acoustic sensor was placed next to the door entrance at an height of approximately 1.2 meters. The two small rooms indicated by '*Room A/B*' were not used during the recordings.

5.5.2 Training and test set generation

In total four independent folds were generated for the experiments of this chapter. The training and test of each fold are obtained by randomly sampling instances from the dataset, i.e. the SAD triggered $\mu\sigma$-MFCC features as discussed in Section 5.2, with a ratio of 75% and 25% respectively. Next, the training sets are balanced to an equal number of presence and absence instances in order to eliminate the potential influence of class imbalance during training. The balancing operation is performed by randomly removing presence samples in the training set. Table 5.3 gives an overview of the amount of presence and absence $\mu\sigma$-MFCC feature frames per fold before balancing. Furthermore, the data within each fold is normalised such that the training set has zero mean and unit standard deviation in each feature dimension.

Table 5.3: Overview of the number of presence and absence $\mu\sigma$-MFCC feature frames per fold before balancing the training sets.

Fold	**Training set**		**Test set**	
	N_{pres}	N_{abs}	N_{pres}	N_{abs}
1	479364	13240	159666	4531
2	479156	13448	159874	4323
3	479160	13444	159870	4327
4	479177	13427	159853	4344

Note: The average amount of presence and absence frames in the training set (before balancing) and test set is 97% and 3%.

5.5.3 Evaluation metric

The used evaluation metric for tuning the hyperparameters and to analyse the obtained results is based on the '*true positive rate*' (tpr) and '*false positive rate*' (fpr). More specifically, the metric is defined by

$$d = \sqrt{fpr^2 + (1 - tpr)^2}, \tag{5.47}$$

and needs to be minimised. The evaluation metric becomes 0 when all samples are classified correctly and $\sqrt{2}$ when all samples are misclassified. The motivation behind this score is that we want to find a solution such that Euclidean distance to the optimal condition, i.e. $tpr = 1$ and $fpr = 0$, is minimised. Furthermore, the metric also equally weighs the importance of tpr and fpr. This implies

that a reduction of $tpr - \Delta$ is equally penalised with an increment of $fpr + \Delta$. Reason for equal weighting is that missed detections causes considerable user inconvenience, e.g. turning off the L-HVAC system when someone is still present in the environment, while a large number of false person detections leads to lower energy savings.

5.5.4 Learning strategies

This section discusses the three different learning strategies that have been investigated during the experiments of this chapter. The first learning strategy basically estimates all model parameters in an offline setting and the obtained results will be considered as a baseline score. The second learning strategy initialises the model architecture defined by $\mathbf{X}_{pv}$ in advance and updates only the parameter vector using recursive least-squares resulting in a fixed-model architecture. The third learning strategy starts with an empty set of prototypes and updates both the model architecture and the parameter vector recursively from the incoming data. Specific details about all three learning strategies are given below.

5.5.3.1 Offline learning

The offline learning strategy evaluates SVM, FS-LSSVM and SR-LSSVM for the purpose of acoustic presence detection. The relation between the amount of training data and presence detection performance is investigated by offline increasing the number of training examples and retraining the model parameters from scratch as shown in Figure 5.3. The hyperparameters, i.e. RBF-kernel bandwidth and regularisation parameter, are tuned on all training data and are kept fixed for all smaller batch sizes. The optimal hyperparameters are determined such that the evaluation metric $d = \sqrt{fpr^2 + (1 - tpr)^2}$ is minimised on a separate development set which was obtained by randomly drawing instances from the training set. The total number of prototype vectors, further denoted by M, for FS-LSSVM and SR-LSSVM is set equal to the number of support vectors learned by the corresponding SVM model. The prototype vectors were defined by randomly sampling from the training set.

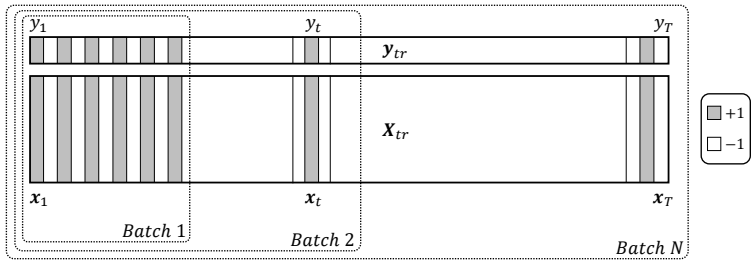

Figure 5.3: Graphical representation of the offline learning strategy. The relation between the amount of training data and presence detection performance is investigated by increasing the number of training examples offline and retraining the model parameters from scratch. Note that each batch has a balanced number of presence and absence samples.

5.5.3.2 Adaptive learning using a fixed-model architecture

The adaptive learning strategy using a fixed-model architecture compares FS-LSSVM and SR-LSSVM when the set of prototypes is initialised in advance by drawing M samples from the entire training set. This learning strategy results in a fixed-model architecture defined by $\mathbf{X}_{pv}$. The parameter vector $\mathbf{w}$ (in case of FS-LSSVM) and $\boldsymbol{\beta}$ (in case of SR-LSSVM) is updated recursively every time a new input pair $(\mathbf{x}_t, y_t)$ is presented to the algorithm by using the normal step RLS updates given in Section 5.4.1. The influence of the number of prototype vectors in $\mathbf{X}_{pv}$ is examined regarding presence detection performance and compared to the results obtained with offline learning. The used hyperparameters are the same as those obtained by the offline learning strategy.

5.5.3.3 Adaptive learning using an adaptive-model architecture

The adaptive learning strategy using an adaptive-model architecture evaluates SR-LSSVM when both the model architecture and parameter vector are updated from the incoming data. This learning strategy allows the model architecture to adapt over time resulting in an adaptive-model behaviour. More specifically, the model starts with an empty set of prototypes and updates both $\mathbf{X}_{pv}$ and $\boldsymbol{\beta}$ from $(\mathbf{x}_t, y_t)$ using the updates of the normal, growing and pruning step as discussed in Section 5.4. The relation between the maximum number of prototypes (M) and the influence of the threshold τ to decide whether $\mathbf{x}_t$ must be incurred as a prototype or not will be investigated regarding presence detection performance.

5.6 Results

5.6.1 Offline learning

The offline learning strategy evaluates SVM, FS-LSSVM and SR-LSSVM when all model parameters are estimated in batch mode and thus no recursive updates are used. The batch sizes are increased with 500 presence and 500 absence samples per iteration to examine the relation between the amount of training data and presence detection performance. The optimal hyperparameters obtained by the gridsearch using all training data are $\lambda = 20$, with $\lambda = 1/2\sigma^2$ as the RBF-kernel parameter, $C = 5$ for SVM and $\gamma = 10$ for FS/SR-LSSVM. The number of prototype vectors in $\mathbf{X}_{pv}$ for FS/SR-LSSVM is set equal to the number of support vectors learned by the corresponding SVM model and are randomly sampled from the training batch.

The relation between batch size vs. presence detection performance is also investigated for three different time sequences: i.e. '*random*', '*normal*' and '*reverse*' sequence. The random time sequence shuffles all training examples such that the time context in the data is lost. The normal time sequence keeps the order as recorded while the reverse time sequence flips the data over time. The motivation behind the different time sequences is to investigate the possible influence of the time context in the data on the presence detection performance.

All offline results are shown in Figure 5.4 in function of the evaluation metric (d) and the number of support vectors/prototypes. These results clearly indicate that SVM, FS-LSSVM and SR-LSSVM achieve similar presence detection scores. It is worth to note that FS-LSSVM and SR-LSSVM even achieve the exact same results due to their equivalent formulation, i.e. $\boldsymbol{\beta} = \boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}}\mathbf{w}$. Table 5.4 gives an overview of the corresponding tpr's and fpr's for a batch size of 1000 (smallest batch), 13000 (middle batch) and 27000 samples (largest batch). As expected, the best scores are obtained when all training data is used to learn the model parameters resulting in a tpr of 95% and a fpr of 6% for all three classifiers but it comes with a rather high number of 5755 support/prototype vectors as one can see in Figure 5.4.

Furthermore, the following two observations can also be made by comparing the obtained results of the three different time sequences. First, randomly shuffling the training data benefits the presence detection performance for smaller batch sizes. The latter is a result of an improved generalisation between the training and test set since the test set contains samples from the whole recording period. However, shuffling the data cannot be done in a practical implementation since the data will be presented sequentially to the learning algorithm. Secondly, the normal time sequence results in an increasing evaluation metric in the region

between 7000 and 12000 training samples. During this period renovation works were in progress in the building next to the office environment resulting in a specific type of recurring background noise, i.e. drilling sounds, and thereby tailoring the classifier to model absence of this type of background sounds. Flipping the data over time yield that renovating sounds occurs at the end and thereby having less influence on the overall presence detection performance.

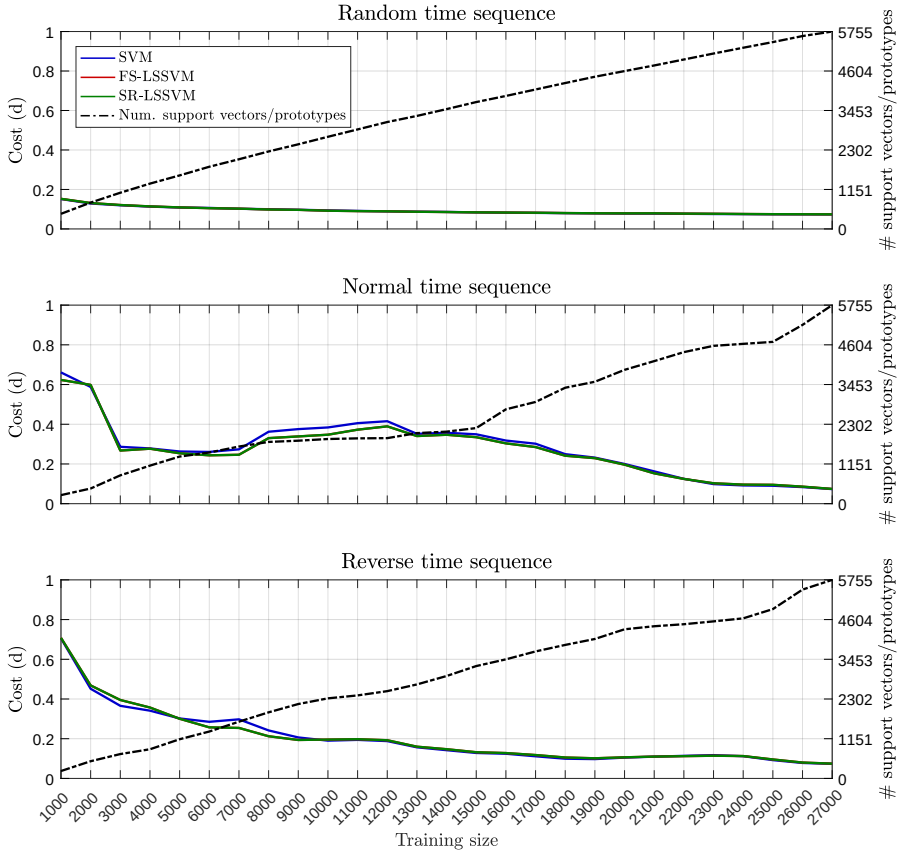

Figure 5.4: The offline presence detection results for the three different time sequences. The obtained results are expressed in function of the evaluation metric (d) and the number of support vectors/prototypes (M).

Table 5.4: The obtained tpr's and fpr's in the offline learning mode for the smallest, middle and largest batch size.

Offline learning		Batch size 1000		13000		27000	
		tpr	fpr	tpr	fpr	tpr	fpr
Random	SVM	92.3	13.0	94.8	7.0	95.2	5.6
	FS-LSSVM	91.1	12.4	94.7	6.9	95.3	5.7
	SR-LSSVM	91.1	12.4	94.7	6.9	95.3	5.7
Normal	SVM	78.6	62.4	97.4	35.1	95.2	5.6
	FS-LSSVM	75.2	56.9	96.6	33.9	95.3	5.7
	SR-LSSVM	75.2	56.9	96.6	33.9	95.3	5.7
Reverse	SVM	68.5	63.1	85.1	4.9	95.2	5.6
	FS-LSSVM	62.8	60.2	85.2	6.0	95.3	5.7
	SR-LSSVM	62.8	60.2	85.2	6.0	95.3	5.7

Note: The same results of FS-LSSVM and SR-LSSVM are due to their equivalent formulation.

5.6.2 Adaptive learning using a fixed-model architecture

The adaptive learning strategy using a fixed-model architecture evaluates FS-LSSVM and SR-LSSVM with a fixed number of prototypes. The set of prototype vectors in $\mathbf{X}_{pv}$ is initialised in advance by randomly drawing M samples from the complete training set and are kept fixed during the remainder of the experiment. Only the parameter vector $\mathbf{w}$ (in case of FS-LSSVM) and $\boldsymbol{\beta}$ (in case of SR-LSSVM) will be updated from $(\mathbf{x}_t, y_t)$ using the RLS normal step updates as discussed in Section 5.4.1. The updated model will be evaluated every 1000 updates to investigate the relation between presence detection performance and the amount of seen training data. Furthermore, the influence of the number of prototypes in $\mathbf{X}_{pv}$ is also investigated by setting $M = \{500, 1000, 1500, 2000\}$.

The obtained results are shown in Figure 5.5 and indicate similar trends as with offline learning. Note that the random time sequence is no longer investigated due to its irrelevance for a practical implementation. The corresponding tpr's and fpr's for a training size of 1000, 13000 and 27000 samples are given in Table 5.5. As expected, increasing the number of prototypes in $\mathbf{X}_{pv}$ improves the presence detection performance but comes at the expense of a higher computational complexity. The final obtained tpr's and fpr's are 93.5% and 10.5% for $M = 500$ and can be further improved to 94.6% and 6.9% by setting $M = 2000$. These results are slightly less accurate than those obtained in the offline setting but the fixed-model architecture enables adaptive learning of the model parameters

and gives direct access in controlling the model complexity. Furthermore, the used number of prototype vectors is also significantly lower than with offline learning and thereby speeding up the computations and reducing the model complexity which is beneficial for embedded implementations.

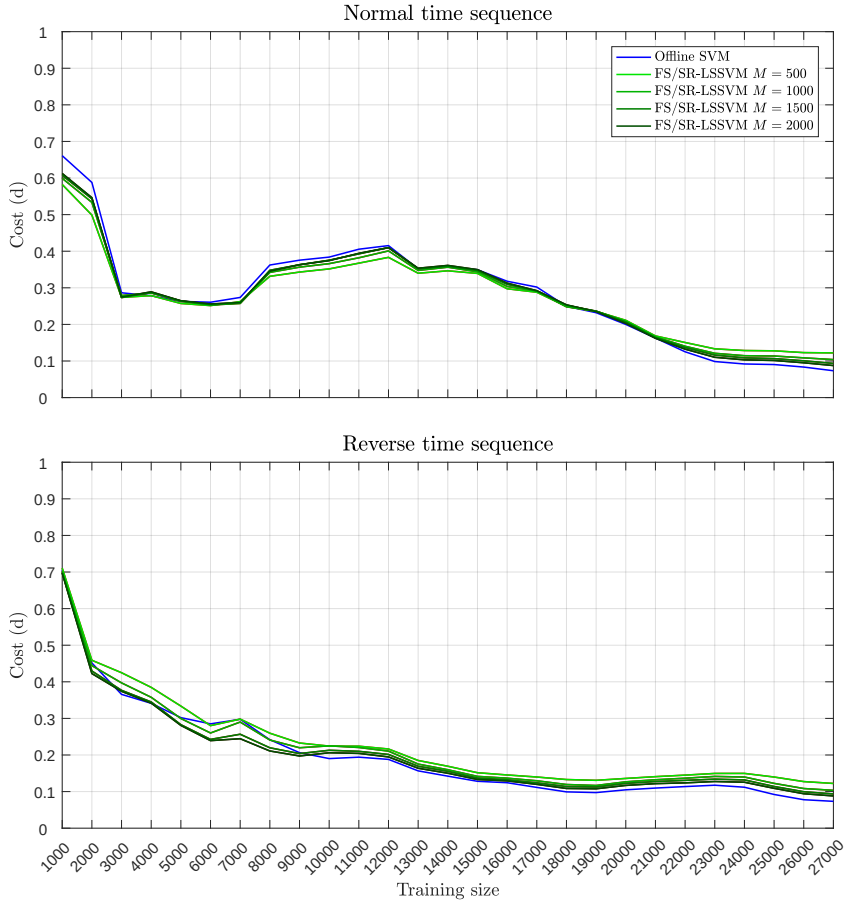

Figure 5.5: The obtained results by the adaptive learning strategy using a fixed-model architecture in function of the training size and the number of prototypes (M) for the normal and reverse time sequence.

Table 5.5: The obtained tpr's and fpr's for the adaptive learning mode using a fixed-model architecture after 1000, 13000 and 27000 training samples.

Adaptive learning fixed-model architecture		**Number of training samples** 1000		13000		27000	
		tpr	fpr	tpr	fpr	tpr	fpr
Normal	$M = 500$	72.1	50.8	97.3	33.9	93.5	10.3
	$M = 1000$	71.8	52.6	96.4	34.6	94.1	8.5
	$M = 1500$	72.3	53.8	96.4	35.0	94.4	7.5
	$M = 2000$	72.5	54.5	96.5	35.2	94.6	6.9
Reverse	$M = 500$	63.7	61.0	88.3	8.0	93.5	10.3
	$M = 1000$	65.1	60.8	84.3	7.8	94.1	8.5
	$M = 1500$	65.6	60.8	84.6	7.1	94.4	7.5
	$M = 2000$	65.8	60.7	85.0	6.6	94.6	6.9

5.6.3 Adaptive learning using an adaptive-model architecture

The adaptive learning strategy using an adaptive-model architecture differs from the previous setting in the sense that $\mathbf{X}_{pv}$ is also learned from the incoming training data. Hence, the updates of the normal, growing and pruning step as discussed in Section 5.4 will be applied to SR-LSSVM. The relation between the amount of training data, the maximum allowable number of prototypes in $\mathbf{X}_{pv}$, and the threshold τ to decide whether the current sample $(\mathbf{x}_t, y_t)$ must be included as prototype will be investigated. The maximum number of prototypes in $\mathbf{X}_{pv}$ is again set to $M = \{500, 1000, 1500, 2000\}$ and the investigated thresholds are $\tau = \{0.1, 0.05, 0.01, 0.005, 0.001, 0.0005\}$.

Figure 5.6 visualises the influence of the threshold τ on the presence detection performance of SR-LSSVM when $M = 500$ and $M = 2000$. From these results it can be clearly observed that τ has a significant impact on the behaviour of the model. Too low values of τ yield a rapid adaptation of $\mathbf{X}_{pv}$ but comes with the risk of overfitting or model drift. This can be clearly observed by the increasing cost (d) occurring after a certain number of training samples for the settings $\tau < 0.05$ when $M = 500$ and $\tau < 0.005$ when $M = 2000$. On the other hand, too high values of τ tend to a very slow adaptation of $\mathbf{X}_{pv}$ causing a suboptimal model architecture. This situation occurs for the setting $\tau > 0.1$ when $M = 500$ and $\tau > 0.05$ when $M = 2000$ and can be observed by the very slowly increasing number of prototypes. As a result, the threshold τ must be carefully chosen and depends on the maximum number of prototypes that $\mathbf{X}_{pv}$ may contain.

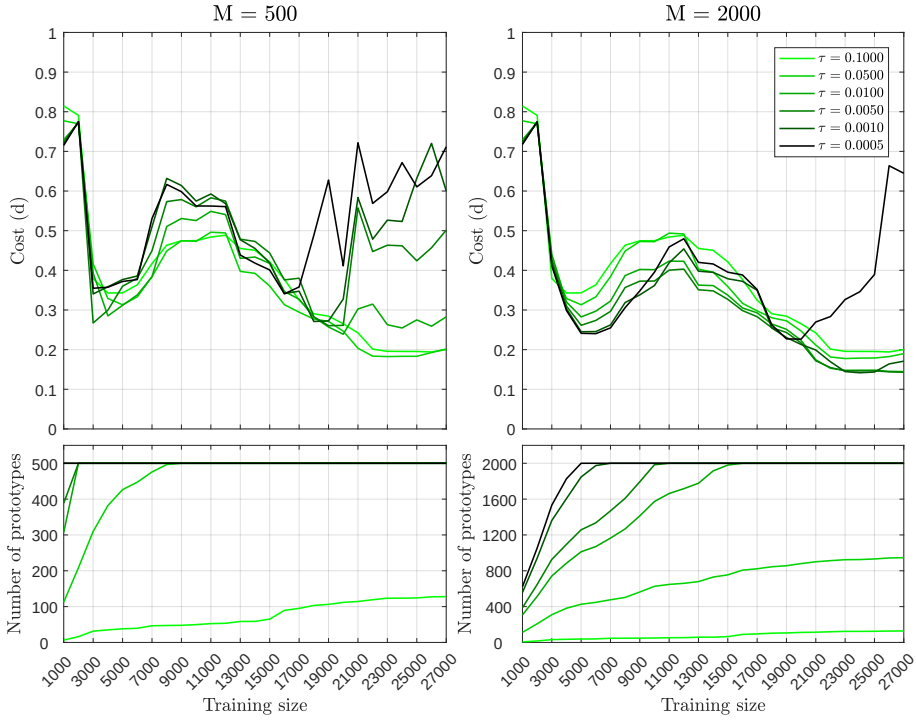

Figure 5.6: The influence of the threshold τ on the presence detection performance of SR-FSLSSVM for $M = 500$ and $M = 2000$ in the normal time sequence.

Figure 5.7 are the obtained results when the threshold τ is set in function of M. More specifically, the used thresholds are $\tau_1 = 0.05$ ($M = 500$), $\tau_2 = 0.01$ ($M = 1000$), $\tau_3 = 0.005$ ($M = 1500$) and $\tau_4 = 0.005$ ($M = 2000$) and are determined experimentally. As one can see, these results indicate the same trends as with previous two learning strategies but the overall scores are less accurate. By examining the tpr's and fpr's given in Table 5.6 it can be observed that the decreased performance is basically the result of increasing false positive rates, i.e. above 10% when all training samples are used, while the true positive rates remain more or less unaffected.

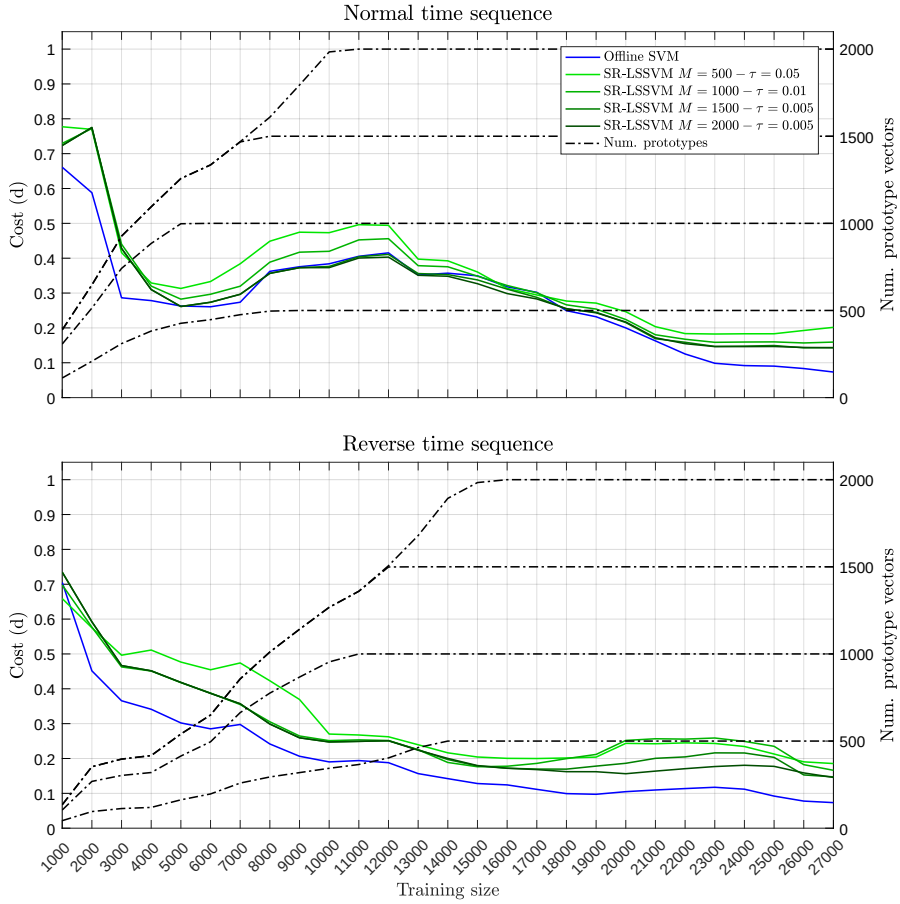

Figure 5.7: The obtained results by the adaptive learning strategy using an adaptive-model architecture in function of the training size and the number of prototypes (M) for the normal and reverse time sequence. The obtained results are expressed in function of the evaluation metric (d) and the number of support vectors/prototypes (M).

Table 5.6: The obtained tpr's and fpr's for the adaptive learning strategy using an adaptive-model architecture after 1000, 13000 and 27000 training samples.

Adaptive learning adaptive-model architecture		**Number of training samples** 1000		13000		27000	
		tpr	fpr	tpr	fpr	tpr	fpr
Normal	$M = 500 - \tau = 0.05$	33.9	40.9	93.1	39.1	82.9	10.3
	$M = 1000 - \tau = 0.01$	42.6	44.8	94.8	37.5	90.2	12.4
	$M = 1500 - \tau = 0.005$	44.2	45.7	94.8	35.2	91.5	11.4
	$M = 2000 - \tau = 0.005$	44.2	45.7	94.3	34.6	92.2	11.8
Reverse	$M = 500 - \tau = 0.05$	39.3	21.9	78.7	10.9	87.8	13.9
	$M = 1000 - \tau = 0.01$	37.6	31.0	79.1	8.4	93.0	15.0
	$M = 1500 - \tau = 0.005$	38.1	39.5	78.8	7.6	91.4	11.8
	$M = 2000 - \tau = 0.005$	38.1	39.5	79.0	7.7	93.5	13.0

5.7 Discussion

As discussed in the previous section, two different adaptive learning strategies have been investigated for the purpose of acoustic presence detection. The first strategy, i.e. adaptive learning using a fixed-model architecture, achieves similar presence detection results as those obtained by the baseline setting where the model parameters are estimated in an offline fashion. The required number of prototype vectors in $\mathbf{X}_{pv}$ for a fixed-model architecture is also significantly lower than the learned number of support vectors by the offline SVM models. Reducing the number of prototypes results in less complex models and requires less memory storage for saving the model parameters. However, the main disadvantage of the fixed-model architecture is that $\mathbf{X}_{pv}$ needs to be initialised in advance from the data. This requires a recording session over a certain amount of time to sample prototypes from the recorded data such that the feature space is well perturbed by $\mathbf{X}_{pv}$.

The second learning strategy, i.e. adaptive learning using an adaptive-model architecture, starts with an empty set in $\mathbf{X}_{pv}$ and enables incrementally learning the prototype vectors and model weight vector from the data. Hence, each time a new input pair $(\mathbf{x}_t, y_t)$ is presented to the algorithm, the updates of the normal, growing and pruning step will be evaluated. This has as advantage that no initialisation of $\mathbf{X}_{pv}$ is required but it comes at the cost of less precise presence detection results due to increased false positive rates. Additional research will be performed to further improve the adaptive-model architecture

results as well. A possible solution is the use of a variable threshold instead of a fixed one or the use of a variable criterion to determine whether $\mathbf{x}_t$ must be included as prototype or not. More specifically, at start up it is desired that input samples can be added rapidly to the set of prototypes while over time a more conservative behaviour in $\mathbf{X}_{pv}$ is preferred such that only those samples are added which really contribute to the solution. Another possible strategy is the use of a '*generic*' set of prototypes learned from multiple environments as initialisation instead of starting from an empty set.

Another research path that already has been started is focussing on implementing the proposed adaptive learning strategy on an ARM Cortex-M7 embedded platform. Algorithmic trade-off's will be investigated to reduce the computational load and to speed up the computations to enable local processing. It has been shown that the sampling frequency is the most dominant parameter in defining the computational complexity of the acoustic pre-processing stage, and the number of prototype vectors in combination with the feature dimension for the local on-edge classifier. Hence, the influence of reducing these parameters on the presence detection performance will be investigated. The used data precision format is also an important factor in defining the computational complexity and the required memory allocation for storing the model parameters. All computations in this chapter are done using a 64-bit floating point data representation which is typically not supported by a digital signal processor (DSP) or a microcontroller unit (MCU). Hence, the use of a less precise and fixed-point data format (e.g. 8/16/32 bit) on the presence detection performance will also be investigated. Furthermore, the use of approximated functions to replace their computational expensive and time consuming counterparts such as logarithms and exponents will also be investigated [70].

5.8 Conclusion

This chapter is devoted to the development of an on-edge and adaptive acoustic presence detection algorithm. The used learning strategy is based on a least-squares support vector machine (LS-SVM) which has some interesting advantages compared to a classical SVM setup. First, the least-squares optimisation yields a solution defined by a set of linear equations instead of a quadratic programming problem [161]. Solving such systems can easily be done locally on an embedded platform using already available optimised microcontroller unit libraries. However, standard LS-SVM formulations are known to not result in a sparse solution and thus becoming very expensive to solve in case of large-scale datasets. In this work two approximation schemes are investigated to enforce a sparse solution and are known by fixed-size LS-

SVM (FS-LSSVM) [45] and subset-of-regressors LS-SVM (SR-LSSVM) [79, 150]. Secondly, the adaptive learning aspect can be easily introduced by using recursive least-squares (RLS) updates. Hence, an update of the model parameters is done each time a new input sample is presented to the algorithm. This approach in adapting the model parameters has as advantage that no data must be stored locally. Two different adaptive learning schemes are examined in the experiments of this chapter. The first scheme involves a fixed-model architecture, i.e. the set of prototype vectors in $\mathbf{X}_{pv}$ is initialised in advance, and only the model weight vector is updated recursively from the incoming data. The second scheme updates both the model architecture $\mathbf{X}_{pv}$ and the weight vector from the incoming data resulting in an adaptive-model architecture behaviour.

All experiments were performed on a real-life acoustic dataset which was recorded at an office environment especially for the development and evaluation of an on-edge and adaptive presence detection classifier. In total 555 hours of acoustic data, i.e. 105 hours of presence and 450 hours of absence, was recorded over a period of 28 days and was labelled on-the-fly by means of a simple button press when entering or leaving the office. The obtained results indicate that adaptive learning using a fixed-model architecture achieves similar presence detection scores compared to offline learning (baseline results). The final presence detection scores when the number of prototypes in $\mathbf{X}_{pv}$ is set to $M = 500$ yield a true positive rate (tpr) of 93.5% and a false positive rate (fpr) of 10.3%. Increasing the number of prototypes to $M = 2000$ further improves the presence detection performance to a tpr of 94.6% and a fpr of 6.9%. The adaptive learning strategy using an adaptive-model architecture enables to learn the set of prototypes from the input data as well, i.e. it does not require to define $\mathbf{X}_{pv}$ in advance, but it comes with the cost of less accurate results. More specifically, by comparing the obtained results with those from the fixed-model architecture it can be noticed that the decrease in performance is basically the result of increasing fpr's. The final obtained results for $M = 500$ are a tpr of 82.9/87.8% and a fpr of 10.3/13.9% for the normal and reverse time sequence respectively. Increasing the number of prototype vectors to $M = 2000$ result in a tpr of 92.2/93.5% and a fpr of 11.8/13.0% for both time sequences respectively.

Chapter 6

Conclusion and future work

This chapter reviews the most important results of this doctoral dissertation and discusses some interesting future research directions.

6.1 Summary of results

This doctoral research is devoted to the use of acoustics as sensing modality for indoor monitoring applications by taking into account the following four challenges as discussed in Section 1.5. The first challenge focusses on the use of '*simple*' and '*resource efficient*' models which can be embedded locally on today's off-the-shelf mobile devices. Local processing of the data is known by '*machine learning on the (extreme) edge*' and has as main advantage that no sensitive data must be transmitted and thus comes with an improved user's privacy. The second challenge is devoted to the use of a '*weakly-supervised*' and '*multi-label*' learning strategy. Weak supervision is defined by high level and less accurate labelling information which might not entirely correspond to the ground truth but are cheaper to obtain. The multi-label aspect comes from the situation that a larger sound segment can contain multiple events from different sound classes. The third challenge is devoted to the ability of the learning algorithm to deal with '*overlapping sound events*' which also typically occur in real-life acoustical data. The last challenge focusses to the use of an '*adaptive*' learning strategy. The aspect of adaptive learning implies that the model parameters can be updated in a stepwise manner from small batches of newly arrived samples only. The latter reduces the required memory of the embedded platform and allows to model slowly varying changes in the data which is a desirable property in case of real-life settings.

As already mentioned in the introduction of this dissertation, the main contributions of this doctorate are categorised into three main research topics each tackling one (or more) of the above mentioned challenges. The final concluding remarks are:

- **Part I - supervised classification of activities of daily living** (Challenge I) is devoted to the use of a wireless acoustic sensor network (WASN) for the purpose of '*supervised classification*' of activities of daily living (ADL). A WASN has as main advantage that a large spatial sampling of the home environment can be obtained without any inconvenient cabling among the acoustic sensor locations for synchronisation and data transfer. The computational complexity of the proposed system architecture is examined in runtime mode for different parameter settings and is evaluated w.r.t. classification accuracy and the robustness to background noise. The used acoustic features are the MEL-frequency cepstral coefficients (MFCC) and the two examined ML classifiers are a Gaussian mixture model (GMM) and a support vector machine (SVM). A real-life dataset containing ten different activities has been recorded by a WASN which was installed in a

domestic environment and all recorded data was labelled manually on the activity level to speed up the annotation process.

The obtained results indicate that GMM and SVM achieve a statistically equivalent classification score when the sampling frequency is set to 32 kHz. The corresponding results are $76.5 \pm 4.8\%$ for GMM and $78.2 \pm 4.1\%$ for SVM. However, the main advantage of SVM in this work is that it uses the cepstral mean and standard deviation computed on a 30 second window basis (corresponding to the minimum activity duration) as input instead of the raw MFCC features as with GMM. This implies that only one feature vector must be classified every 30 seconds making SVM from a computational point of view more attractive than GMM. Lowering the sampling frequency from 32 kHz to 16 kHz and reducing the number of MEL-filters from 20 to 15 has no affect on the classification performance of SVM, but it comes with a reduced computational complexity in the feature extraction stage of a factor 2.43 in terms of arithmetic operation cycles. Further decreasing the feature configuration parameters to speed up the computations results in statistical significant lower scores.

The robustness of the WASN to background noise compared to a single microphone solution is also examined during the experiments. The obtained results indicate that the dynamic microphone selection by the WASN yield higher classification accuracies for medium and high SNRs. The average increase in classification accuracy is situated between 0.8% and 4.0% compared to a single microphone setup. For very low SNRs dynamically selecting the best microphone no longer yield an improved classification score. On the other hand, it is also observed that in noise free conditions one microphone per room is typically sufficient for ADL classification.

- **Part II - weakly supervised convolutive event modelling and classification** (Challenge II & III) introduces an extension to non-negative matrix deconvolution (NMD) such that the model parameters, i.e. the time-frequency dictionary elements, can be learned from acoustic data which is supported by '*low-resolution and multi-label*' (LRM) labelling information. The LRM information simply indicates the (subset of) acoustic events that take place over a longer period of time without identifying beginning nor endings. This approach in labelling the data speeds up the annotation process significantly. The existing NMD is extended by incorporating the LRM information into the learning procedure to obtain an enhanced model representation. In the experiments of this chapter we have evaluated LRM-NMD in two different operating modes, i.e. '*single-label learning*' and '*low-resolution multi-label learning*', and have investigated the influence of the amount of labelling information,

background noise and overlapping events (in case of multi-label learning) with respect to classification accuracy. A publicly available dataset of domestic sound events was used to evaluate LRM-NMD [76, 77]. The used features are the MEL-magnitude spectrograms.

The single-label learning mode evaluates LRM-NMD when all events are labelled individually to benchmark our proposed algorithm with other state-of-the-art supervised learning algorithms. The obtained classification results indicate that LRM-NMD (with exception of a supervised NMD setting) outperforms the baselines in situations when the amount of training examples per sound class (n_{tr}) is rather limited, i.e. less than five examples per class, and are situated in the range of 81% and 91% when $n_{tr} = 1$ and $n_{tr} = 5$ respectively. Furthermore, the robustness of LRM-NMD to stationary environmental background noise has been shown till SNRs of 10 dB without any significant loss in classification performance. Lowering the SNRs further yield decreasing scores, but it has been shown that the absolute reduction in classification accuracy is strongly related to the number of training examples per sound class.

The multi-label learning mode evaluates LRM-NMD when the acoustic data is supported by LRM labelling information. The data is presented to the algorithm in segments and the relation between the number of labelled events per segment (n_{lbl}) is examined with respect to classification accuracy. The obtained results indicate that LRM-NMD outperforms the baselines. The corresponding classification accuracies are 75% and 90% when $n_{lbl} = 1$ and $n_{lbl} = 5$ respectively. The robustness to background noise has been shown again with SNRs of down to 10 dB without any significant decrease in classification performance. We have also shown that LRM-NMD can deal with overlapping events in the segments as well, i.e. a drop in absolute classification performance of somewhere around 5% when $n_{overlap} = 50\%$. Further increasing $n_{overlap}$ yields larger declines in classification scores as a result of a higher complexity in the acoustic data.

- **Part III - on-edge adaptive acoustic presence detection** (Challenge I & IV) is devoted to the development of an adaptive learning strategy which can discriminate presence from absence related sounds, i.e. a binary scene classification task. The envisioned machine learning strategies are chosen by taking into account the '*computational complexity*' of the model and the ability to modify the learning strategy to enable '*adaptive*' learning, i.e. such that the model parameters can be efficiently updated from the newly collected samples only without the need of a complete retraining procedure. From this perspective, we have selected a least-squares support vector machine (LS-SVM) based framework. A LS-SVM preserves the SVM methodology but introduces a simplification

such that optimality is defined by a set of linear equations. Linear systems are known to be straightforward to solve, relative simple to embed locally and can easily be modified into a an adaptive variant using recursive least-squares (RLS) updates. Two approximation schemes are examined during the experiments of this chapter to enforce sparse solutions and are known by fixed-size LS-SVM (FS-LSSVM) and subset-of-regressors LS-SVM (SR-LSSVM).

The experiments of this chapter focus on the development of a adaptive learning strategy and envisions two different operating schemes. The first scheme involves a '*fixed-model architecture*', i.e. the set of prototype vectors are initialised in advance by randomly sampling from the data, and only the model weight vector will be updated recursively from the incoming data. The second scheme starts with an empty set of prototypes and updates both the model architecture and the weight vector resulting in an '*adaptive-model architecture*' behaviour. All obtained results are compared to an offline learning setting which is considered as baseline and are investigated w.r.t. the number of seen training samples. A real-life acoustic dataset was recorded at an office environment over a 28 day time span resulting in 555 hours of data which was labelled on-the-fly by the users when entering or leaving the room by means of a simple button push.

The obtained results indicate that adaptive learning using a fixed-model architecture achieves similar presence detection scores as with offline learning. The final presence detection scores when the number of prototypes in the model architecture is set to $M = 500$ yield a true positive rate (tpr) of 93.5% and a false positive rate (fpr) of 10.3%. Increasing the number of prototypes to $M = 2000$ further improves the presence detection performance to a tpr of 94.6% and a fpr of 6.9% but results in an increased computational complexity. The adaptive learning strategy using an adaptive-model architecture enables to learn the set of prototypes from the input data as well, i.e. it does not require to define model architecture in advance, but it comes at the cost of less accurate results. More specifically, by comparing the best obtained results with those from the fixed-model architecture it can be noticed that the decrease in performance is the result of increasing fpr's. The last obtained results for $M = 500$ are a tpr of 82.9/87.8% and a fpr of 10.3/13.9% for two different time sequences, i.e. a normal and reverse time sequence. Increasing the number of prototype vectors to $M = 2000$ result in a tpr of 92.2/93.5% and a fpr of 11.8/13.0% for both time sequences respectively.

6.2 Future research

This section discusses the future research work and will mainly concentrate on following two aspects, i.e '*research towards scientific challenges*' and '*research towards practical challenges*'. The research towards scientific challenges will focus on further optimising the envisioned learning algorithms regarding their shortcomings that have been observed during the experiments of this doctoral research. The research towards practical challenges on the other hand will focus on investigating the implementation and algorithmic trade-offs that must be made to enable a real-time local deployment on a microcontroller unit (MCU). Also the development of real-life demonstrators to valorise the proposed learning algorithms will be considered as a practical research challenge. A more elaborate discussion of both aspects is given in the following two sections.

6.2.1 Research towards scientific challenges

Although the promising results of LRM-NMD still some further scientific research is required for a real-life deployment. The future work will mainly focus on (i) an enhanced initialisation of the dictionary elements, (ii) online updating and augmenting of the dictionary elements from newly collected samples and (iii) labelling on the activity level instead of the event level.

During the experiments of Chapter 4, the set of dictionary elements in $\mathbf{A}^{[o]}$ was always initialised with one example per sound event class from the used dataset. However, the number of sound classes is typically unknown beforehand in a real-life setting making exemplar based initialisation rather impractical. Furthermore, in case of large intra-class variabilities, e.g. door knock on a wooden door versus a composite door, it is also expected that multiple dictionary elements are required to model the same event class. Previous two aspects result in the need of a learning strategy where the model starts from a generic set of dictionary elements, i.e. learned from multiple home environments and representing the most commonly occurring domestic sound events, and which can be adapted to the actual situation given newly collected samples. The latter can be done by retraining the model parameters, i.e. the set of dictionary elements in $\mathbf{A}^{[o]}$ and the corresponding labelling matrix $\mathbf{A}^{[s]}$, using the proposed multiplicative update formulas. It will also be investigated to modify the learning algorithm such that $\mathbf{A}^{[o]}$ can be augmented with additional sound events which are not yet represented by $\mathbf{A}^{[o]}$. A possible solution to determine whether an additional element must be added to $\mathbf{A}^{[o]}$is known by automatic relevance determination (ARD) and is already successfully employed in Bayesian Principal Component Analysis (PCA) and sparse Bayesian learning [165, 135].

Another interesting research path of LRM-NMD is the use of acoustic data labelled on the activity level instead of the event level. An activity is basically a sequence of potentially overlapping isolated sound events and each activity is characterised by its own specific combination of recurring sound events. For instance, '*brushing teeth*' contains data from the sound class '*tap water*' in combination with the sound of a '*tooth brush*' while the activity '*shaving*' contains also data of '*tap water*' but this time in combination with the sound of a '*razor*'. This implies that some dictionary elements in $\mathbf{A}^{[o]}$ must be linked to multiple activities while others are simply linked to one specific activity. The latter can be done by initialising the labelling matrix $\mathbf{A}^{[s]}$ with small positive values and let the algorithm learn the relation between the individual sound events and the corresponding activities.

Learning a classifier on top of the activations using a task-driven learning (TDL) strategy as proposed in [14, 13, 15] will also be investigated in future research. Task-driven non-negative matrix factorisation (TNMF) jointly learns a non-negative dictionary and a classifier that minimises the reconstruction error and the classification cost in a common problem. Hence, the dictionary of the TNMF model is encouraged to explain the discriminative information in the data while keeping a low reconstruction cost and thereby boosting the classification problem.

The on-edge adaptive acoustic presence detection as discussed in Chapter 5 also requires additional research. Future work will mainly focus to the use of a generic set of prototypes, i.e. again learned from multiple environments, as a starting point instead of randomly sampling from the data (cfr. fixed-model architecture) or with an empty set and by recursively adding input samples which contribute to the solution (cfr. adaptive-model architecture). Having a good initialisation of $\mathbf{X}_{pv}$ will result in an improved presence detection performance at start up and thereby requiring less examples to adapt the model to the current environment. Also the use of an adaptive threshold or criterion to decide whether a newly collected input sample must be included as prototype or not will be investigated. During the experiment we have used a fixed threshold which was set manually in function of the maximum allowable number of prototypes in $\mathbf{X}_{pv}$. Such a strategy is rather impractical in case of a real-life setting since it needs to be tuned manually. A possible solution to overcome this problem is by taking the number of seen training samples into account for defining the threshold or criterion. More specifically, at start up it is desired that input samples can be added rapidly to the set of prototypes while over time a more conservative behaviour in $\mathbf{X}_{pv}$ is preferred such that only those samples are added which really contribute to the solution.

Another aspect that will be investigated is the use of a deep learning solution for the three main research contributions of this doctoral dissertation, i.e. ADL

classification, weakly-supervised event modelling and classification, and adaptive presence detection. Research in the scientific community has shown that deep learning typically outperforms the traditional machine learning strategies and have thereby become the new state-of-the-art [159, 176, 130]. Recent advances in today's off-the-shelf mobile embedded platforms starts enabling deploying deep learning solutions locally as well. For instance, the authors of [92] present a real-time convolutional neural network (CNN) for image recognition which was embedded on an ARM Cortex-M7.

6.2.2 Research towards practical challenges

Reducing the overall computational complexity of the envisioned machine learning algorithms will be the main practical research challenge. Both the implementation and algorithmic specific trade-offs will be investigated to enable a local deployment on an embedded platform. For instance, many of today's microcontroller units (MCU) or digital signal processors (DSP) have an optimised instruction set for a fixed-point representation. Hence, the influence of using a less-precise and fixed-point data format, i.e. 8/16-bit fixed-point instead of 64-bit floating-point, or the use of approximated functions to replace the cost expensive algorithmic operators, e.g. logarithms, exponentials, etc., needs to be investigated. Nowadays, there are already optimised toolboxes publicly available on the internet, e.g. the ARM Cortex Microcontroller Software Interface Standard (CMSIS) which are developed for the ARM Cortex M-series, enabling rapid prototyping of the machine algorithms [92].

On the other hand, reducing the complexity on the algorithmic side will also be investigated. For instance, despite the straightforward formulation of LRM-NMD it is still computational expensive to solve due to the iterative approach. Research will focus in optimising the proposed multiplicative updates for LRM-NMD. A possible strategy is by starting the multiplicative updates using all model parameters, i.e. all learned dictionary elements, and discarding those not contributing to the solution during the next iterations. This implies that the number of model parameters will be reduced resulting in fewer computations. Furthermore, it is also expected that a 'good' generic initialisation of $\mathbf{A}^{[o]}$ will result in a lower number of iterations until convergence is reached and thereby speeding up the computations as well.

The development of demonstrators to valorise the proposed learning algorithms of this doctoral dissertation will be considered as the last practical research challenge. Having the ability of demonstrating real-life showcases of its potential to (start-up) companies, spin-offs or other research groups may result into either new fundamental projects or product development ventures. Two interesting use

cases that will be developed are (1) an embedded implementation of LRM-NMD to show its capacity of automatically learning and classifying of events from acoustic data supported by weakly-supervised and multi-label information and (2) a real-time and adaptive human occupancy detector using an enhanced embedded adaptive presence detection strategy.

Chapter 7

Appendices

7.1 LRM-NMD multiplicative updates

This section describes the derivation of the multiplicative updates for LRM-NMD. First, let us briefly recall the used notations:

- $\mathbf{Y}^{[o]} \in \mathbb{R}_{\geq 0}^{B \times F}$ and $\mathbf{\Psi}^{[o]} \in \mathbb{R}_{\geq 0}^{B \times F}$ as the observed and reconstructed input spectrogram,
- $\mathbf{y}^{[s]} \in \mathbb{R}_{\geq 0}^{C}$ and $\boldsymbol{\psi}^{[s]} \in \mathbb{R}_{\geq 0}^{C}$ as the ground truth and reconstructed labelling vector,
- $\mathbf{A}_t^{[o]} \in \mathbb{R}_{\geq 0}^{B \times L}$, with $t \in [1, T]$, as the set of dictionaries at time index t,
- $\mathbf{A}^{[s]} \in \mathbb{R}_{\geq 0}^{C \times L}$ as the labelling matrix of $\mathbf{A}^{[o]}$,
- $\mathbf{X} \in \mathbb{R}_{\geq 0}^{L \times F}$ as the activation matrix,
- and $\mathbf{1}_{\langle F \rangle}$ an all-one column vector of length F.

The cost function of LRM-NMD is defined by,

$$J(\mathbf{A}^{[o]}, \mathbf{A}^{[s]}, \mathbf{X}) = D_{KL}(\mathbf{Y}^{[o]} \| \sum_{t=1}^{T} \mathbf{A}_t^{[o]} \overrightarrow{\mathbf{X}}^{(t-1)}) + \eta D_{KL}(\mathbf{y}^{[s]} \| \mathbf{A}^{[s]} \mathbf{X} \mathbf{1}_{\langle F \rangle}) + \lambda \|\mathbf{X}\|_1, \tag{7.1}$$

with $D_{KL}(V \| W)$ denoting the Kullback-Leibler divergence between V and W. Here, we will use gradient descent as proposed in [102] to obtain multiplicative updates for $\mathbf{A}^{[o]}$, $\mathbf{A}^{[s]}$ and $\mathbf{X}$ for minimising Equation 7.1.

7.1.1 Activations (X)

Taking the gradient of Equation 7.1 with respect to element x_{mn} gives:

$$\begin{aligned}\frac{\partial J(\mathbf{A}^{[o]}, \mathbf{A}^{[s]}, \mathbf{X})}{\partial x_{mn}} = & \frac{\partial}{\partial x_{mn}} \Big[\sum_b \sum_f \sum_t \sum_l a_{t,bl}^{[o]} x_{l,f-t+1} - \sum_b \sum_f y_{bf}^{[o]} \ln \sum_t \sum_l a_{t,bl}^{[o]} x_{l,f-t+1} \Big] \\ & + \frac{\partial}{\partial x_{mn}} \Big[\eta \sum_c \sum_l \sum_f a_{cl}^{[s]} x_{lf} - \eta \sum_c y_c^{[s]} \ln \sum_l \sum_f a_{cl}^{[s]} x_{lf} \Big] \\ & + \frac{\partial}{\partial x_{mn}} \Big[\lambda \sum_l \sum_f x_{lf} \Big],\end{aligned}$$

where the first term of the gradient is the contribution of $D_{KL}(\mathbf{Y}^{[o]} \| \mathbf{\Psi}^{[o]})$, the second term of $\eta D_{KL}(\mathbf{y}^{[s]} \| \boldsymbol{\psi}^{[s]})$ and the last term of $\lambda \|\mathbf{X}\|_1$.

Splitting the gradient into a positive and negative part results into:

$$\left[\frac{\partial J(\mathbf{A}^{[o]},\mathbf{A}^{[s]},\mathbf{X})}{\partial x_{mn}}\right]_+ = \sum_b \sum_t a^{[o]}_{t,bm} + \eta \sum_c a^{[s]}_{cm} + \lambda,$$

and,

$$\left[\frac{\partial J(\mathbf{A}^{[o]},\mathbf{A}^{[s]},\mathbf{X})}{\partial x_{mn}}\right]_- = \sum_b \sum_t \frac{y^{[o]}_{b,n+t-1} a^{[o]}_{t,bm}}{\sum_p \sum_l a^{[o]}_{p,bl} x_{l,n+t-p}} + \eta \sum_c \frac{y^{[s]}_c a^{[s]}_{cm}}{\sum_l \sum_f a^{[s]}_{cl} x_{lf}}.$$

We thus obtain the following multiplicative update for x_{mn}:

$$x_{mn} \leftarrow x_{mn} \frac{\sum_b \sum_t \frac{y^{[o]}_{b,n+t-1} a^{[o]}_{t,bm}}{\sum_p \sum_l a^{[o]}_{p,bl} x_{l,n+t-p}} + \eta \sum_c \frac{y^{[s]}_c a^{[s]}_{cm}}{\sum_l \sum_f a^{[s]}_{cl} x_{lf}}}{\sum_b \sum_t a^{[o]}_{t,bm} + \eta \sum_c a^{[s]}_{cm} + \lambda},$$

which can be rewritten in the following matrix form:

$$\mathbf{X} \leftarrow \mathbf{X} \circ \frac{\left(\sum_{t=1}^{T} \mathbf{A}_t^{[o]\top} \left[\frac{\overleftarrow{\mathbf{Y}^{[o]}}^{(t-1)}}{\mathbf{\Psi}^{[o]}}\right]\right) + \eta\left(\mathbf{A}^{[s]\top}\left[\frac{\mathbf{y}^{[s]}}{\boldsymbol{\psi}^{[s]}}\right]\mathbf{1}^\top_{\langle F\rangle}\right)}{\left(\sum_{t=1}^{T} \mathbf{A}_t^{[o]\top} \overleftarrow{\mathbf{1}}^{(t-1)}_{\langle B\times F\rangle}\right) + \eta\left(\mathbf{A}^{[s]\top}\mathbf{1}_{\langle C\rangle}\mathbf{1}^\top_{\langle F\rangle}\right) + \lambda},$$

with $\mathbf{1}_{\langle C\rangle}$ an all-one column vector of length C and $\mathbf{1}_{\langle B\times F\rangle}$ an all-one matrix of dimensions $B \times F$.

7.1.2 Dictionary elements ($\mathbf{A}^{[o]}$)

Taking the gradient of Equation 7.1 with respect to element $a^{[o]}_{u,mn}$ gives:

$$\frac{\partial J(\mathbf{A}^{[o]},\mathbf{A}^{[s]},\mathbf{X})}{\partial a^{[o]}_{u,mn}} = \frac{\partial}{\partial a^{[o]}_{u,mn}}\left[\sum_b \sum_f \sum_t \sum_l a^{[o]}_{t,bl} x_{l,f-t+1} - \sum_b \sum_f y^{[o]}_{bf} \ln \sum_t \sum_l a^{[o]}_{t,bl} x_{l,f-t+1}\right].$$

Splitting the gradient into a positive and negative part results into:

$$\left[\frac{\partial J(\mathbf{A}^{[o]},\mathbf{A}^{[s]},\mathbf{X})}{\partial a^{[o]}_{u,mn}}\right]_+ = \sum_f x_{n,f-u+1},$$

and,

$$\left[\frac{\partial J(\mathbf{A}^{[o]},\mathbf{A}^{[s]},\mathbf{X})}{\partial a^{[o]}_{u,mn}}\right]_- = \sum_f \frac{y^{[o]}_{mf} x_{n,f-u+1}}{\sum_p \sum_q a^{[o]}_{p,mq} x_{q,f-p+1}}.$$

We thus obtain the following multiplicative update for $a^{[o]}_{u,mn}$:

$$a^{[o]}_{u,mn} \leftarrow a^{[o]}_{u,mn} \frac{\sum_f \frac{y^{[o]}_{mf} x_{n,f-u+1}}{\sum_p \sum_q a^{[o]}_{p,mq} x_{q,f-p+1}}}{\sum_f x_{n,f-u+1}},$$

which can be rewritten in the following matrix form:

$$\mathbf{A}_t^{[o]} \leftarrow \mathbf{A}_t^{[o]} \circ \frac{\left[\frac{\mathbf{Y}^{[o]}}{\mathbf{\Psi}^{[o]}}\right] \overrightarrow{\mathbf{X}}^{(t-1)^\top}}{\mathbf{1}_{\langle B \times F \rangle} \overrightarrow{\mathbf{X}}^{(t-1)^\top}}, \forall t \in [1, T],$$

with $1_{\langle B \times F \rangle}$ denoting an all-one matrix of dimensions $B \times F$.

7.1.3 Labelling matrix ($\mathbf{A}^{[s]}$)

Taking the gradient of Equation 7.1 with respect to element $a_{mn}^{[s]}$ gives:

$$\frac{\partial J(\mathbf{A}^{[o]}, \mathbf{A}^{[s]}, \mathbf{X})}{\partial a_{mn}^{[s]}} = \frac{\partial}{\partial a_{mn}^{[s]}} \Big[\sum_c \sum_l \sum_f a_{cl}^{[s]} x_{lf} - \sum_c y_c^{[s]} \ln \sum_l \sum_f a_{cl}^{[s]} x_{lf} \Big].$$

Splitting the gradient into a positive and negative part results into:

$$\left[\frac{\partial J(\mathbf{A}^{[o]}, \mathbf{A}^{[s]}, \mathbf{X})}{\partial a_{mn}^{[s]}}\right]_+ = \sum_f x_{nf},$$

and,

$$\left[\frac{\partial J(\mathbf{A}^{[o]}, \mathbf{A}^{[s]}, \mathbf{X})}{\partial a_{mn}^{[s]}}\right]_- = \sum_f \frac{y_m^{[s]} x_{nf}}{\sum_l \sum_p a_{ml}^{[s]} x_{lp}}.$$

We thus obtain the following multiplicative update for $a_{mn}^{[s]}$:

$$a_{mn}^{[s]} \leftarrow a_{mn}^{[s]} \frac{\sum_f \frac{y_m^{[s]} x_{nf}}{\sum_l \sum_p a_{ml}^{[s]} x_{lp}}}{\sum_f x_{nf}},$$

which can be rewritten in the following matrix form:

$$\mathbf{A}^{[s]} \leftarrow \mathbf{A}^{[s]} \circ \frac{\left[\frac{\boldsymbol{y}^{[s]}}{\boldsymbol{\psi}^{[s]}}\right] \left[\mathbf{X} \mathbf{1}_{\langle F \rangle}\right]^\top}{\mathbf{1}_{\langle C \rangle} \left[\mathbf{X} \mathbf{1}_{\langle F \rangle}\right]^\top},$$

with $1_{\langle C \rangle}$ and $1_{\langle F \rangle}$ denoting an all-one column vector of length C and F respectively.

7.2 LRM-NMD results

7.2.1 Single-label learning mode results

Table 7.1: Classification scores of LRM-NMD in single-label learning mode on the clean data in function of the number of training examples per sound class (n_{tr}), semantic balancing parameter (η) and sparsity penalisation (λ).

λ	η	Number of training examples per sound class (n_{tr})						
		1	**2**	**3**	**4**	**5**	**10**	**all**
0	**0**	$49.1 \pm 2.4\%$	$45.5 \pm 1.6\%$	$41.4 \pm 5.1\%$	$39.4 \pm 3.2\%$	$37.3 \pm 3.4\%$	$37.3 \pm 2.5\%$	$35.6 \pm 2.0\%$
	0.1	$50.8 \pm 3.4\%$	$48.6 \pm 3.2\%$	$47.0 \pm 3.2\%$	$43.9 \pm 4.3\%$	$42.2 \pm 4.3\%$	$40.9 \pm 3.3\%$	$40.2 \pm 2.8\%$
	1	$57.0 \pm 4.9\%$	$59.7 \pm 3.6\%$	$60.0 \pm 4.7\%$	$59.2 \pm 5.3\%$	$59.7 \pm 6.6\%$	$63.9 \pm 3.1\%$	$63.9 \pm 0.8\%$
	10	$57.8 \pm 5.6\%$	$61.3 \pm 2.4\%$	$63.1 \pm 3.3\%$	$63.8 \pm 7.0\%$	$65.8 \pm 7.4\%$	$72.7 \pm 2.4\%$	$71.7 \pm 2.5\%$
	100	$58.8 \pm 5.5\%$	$62.3 \pm 2.9\%$	$64.4 \pm 2.8\%$	$63.8 \pm 4.4\%$	$65.3 \pm 4.6\%$	$72.0 \pm 1.6\%$	$72.3 \pm 1.7\%$
	200	$58.4 \pm 6.5\%$	$63.4 \pm 3.2\%$	$63.9 \pm 3.9\%$	$63.4 \pm 3.6\%$	$65.5 \pm 6.1\%$	$72.5 \pm 2.0\%$	$72.2 \pm 1.9\%$
	500	$58.4 \pm 7.4\%$	$63.4 \pm 3.5\%$	$64.8 \pm 4.8\%$	$64.7 \pm 4.5\%$	$66.1 \pm 4.2\%$	$71.7 \pm 1.9\%$	$73.1 \pm 1.5\%$
	1000	$59.7 \pm 8.3\%$	$64.5 \pm 2.5\%$	$65.8 \pm 3.9\%$	$65.5 \pm 3.2\%$	$66.9 \pm 3.3\%$	$72.2 \pm 1.7\%$	$72.7 \pm 1.5\%$
5	**0**	$73.8 \pm 6.2\%$	$73.8 \pm 3.1\%$	$73.3 \pm 3.7\%$	$69.2 \pm 4.9\%$	$65.8 \pm 5.7\%$	$63.8 \pm 2.1\%$	$61.9 \pm 2.2\%$
	0.1	$74.7 \pm 6.9\%$	$78.6 \pm 5.9\%$	$76.9 \pm 1.8\%$	$73.8 \pm 3.8\%$	$73.0 \pm 4.6\%$	$71.1 \pm 3.1\%$	$70.8 \pm 2.4\%$
	1	$75.2 \pm 6.4\%$	$85.2 \pm 4.2\%$	$83.4 \pm 5.5\%$	$8.50 \pm 5.1\%$	$85.2 \pm 3.8\%$	$89.8 \pm 2.0\%$	$90.6 \pm 2.2\%$
	10	$77.3 \pm 6.9\%$	$85.9 \pm 4.2\%$	$88.3 \pm 1.7\%$	$89.4 \pm 2.2\%$	$90.5 \pm 3.3\%$	$93.8 \pm 0.9\%$	$93.4 \pm 0.6\%$
	100	$81.1 \pm 4.2\%$	$87.8 \pm 3.4\%$	$89.5 \pm 3.5\%$	$91.1 \pm 2.2\%$	$91.3 \pm 2.2\%$	$94.2 \pm 1.1\%$	$93.9 \pm 1.2\%$
	200	$80.2 \pm 3.1\%$	$88.4 \pm 1.9\%$	$88.4 \pm 3.6\%$	$89.7 \pm 2.3\%$	$90.8 \pm 2.4\%$	$93.3 \pm 2.1\%$	$93.6 \pm 1.6\%$
	500	$78.1 \pm 3.6\%$	$85.6 \pm 2.7\%$	$85.5 \pm 4.0\%$	$85.6 \pm 2.6\%$	$84.7 \pm 4.5\%$	$92.2 \pm 3.6\%$	$92.0 \pm 2.7\%$
	1000	$74.1 \pm 4.8\%$	$81.7 \pm 2.8\%$	$80.6 \pm 3.3\%$	$82.2 \pm 3.6\%$	$83.4 \pm 5.2\%$	$88.8 \pm 4.5\%$	$89.5 \pm 3.6\%$
10	**0**	$71.4 \pm 5.3\%$	$69.5 \pm 3.2\%$	$67.7 \pm 3.8\%$	$63.9 \pm 6.1\%$	$61.1 \pm 4.8\%$	$58.9 \pm 4.3\%$	$55.5 \pm 2.1\%$
	0.1	$72.5 \pm 6.7\%$	$74.4 \pm 5.1\%$	$70.3 \pm 2.9\%$	$67.5 \pm 5.2\%$	$67.3 \pm 5.6\%$	$64.5 \pm 5.5\%$	$63.8 \pm 5.5\%$
	1	$75.9 \pm 7.3\%$	$83.9 \pm 3.5\%$	$83.8 \pm 3.1\%$	$83.1 \pm 2.2\%$	$81.6 \pm 4.5\%$	$87.8 \pm 2.8\%$	$88.3 \pm 3.1\%$
	10	$75.9 \pm 6.9\%$	$85.6 \pm 4.4\%$	$88.4 \pm 2.5\%$	$87.8 \pm 2.8\%$	$88.1 \pm 3.2\%$	$92.8 \pm 1.9\%$	$93.6 \pm 1.4\%$
	100	$77.7 \pm 4.5\%$	$85.0 \pm 3.5\%$	$85.5 \pm 3.9\%$	$84.2 \pm 1.4\%$	$84.5 \pm 2.2\%$	$90.6 \pm 0.7\%$	$90.0 \pm 1.4\%$
	200	$73.0 \pm 5.3\%$	$84.1 \pm 3.2\%$	$83.3 \pm 3.9\%$	$82.0 \pm 2.2\%$	$84.8 \pm 1.8\%$	$89.7 \pm 1.1\%$	$89.4 \pm 1.4\%$
	500	$69.8 \pm 6.4\%$	$78.0 \pm 2.8\%$	$75.6 \pm 3.9\%$	$73.9 \pm 2.4\%$	$76.9 \pm 4.0\%$	$86.4 \pm 3.4\%$	$86.6 \pm 1.9\%$
	1000	$66.1 \pm 5.3\%$	$72.0 \pm 3.2\%$	$68.8 \pm 2.8\%$	$67.7 \pm 2.7\%$	$69.7 \pm 6.1\%$	$80.5 \pm 3.2\%$	$80.6 \pm 3.2\%$

Note: the best classification scores for each n_{tr} setting are underlined.

7.2.2 Low-resolution multi-label results

Table 7.3: Classification scores of LRM-NMD in low-resolution multi-label learning mode for $\eta = 5$ and $\lambda = 5$ in function of the number of labelled events per annotating segment (n_{lbl}), signal to noise ratio (SNR) and degree of overlap in the annotating segments ($n_{overlap}$).

$\mathbf{n_{overlap}}$ (in %)	**SNR** (in dB)	**Number of labelled events per annotating segment** (n_{lbl}) **1**	**2**	**3**	**4**	**5**
0	**Clean**	$73.9 \pm 5.3\%$	$82.3 \pm 5.6\%$	$86.7 \pm 2.1\%$	$89.1 \pm 1.2\%$	$90.2 \pm 3.9\%$
	20	$73.6 \pm 5.1\%$	$82.3 \pm 5.3\%$	$86.6 \pm 2.1\%$	$88.6 \pm 1.1\%$	$90.2 \pm 3.4\%$
	10	$73.8 \pm 2.6\%$	$82.0 \pm 4.9\%$	$86.4 \pm 1.9\%$	$87.7 \pm 0.9\%$	$89.5 \pm 3.3\%$
	5	$71.9 \pm 3.6\%$	$78.0 \pm 4.4\%$	$81.3 \pm 3.6\%$	$85.9 \pm 1.1\%$	$88.1 \pm 3.3\%$
	3	$60.8 \pm 2.4\%$	$69.7 \pm 3.9\%$	$73.9 \pm 4.4\%$	$80.2 \pm 2.2\%$	$84.4 \pm 3.1\%$
	0	$40.3 \pm 7.0\%$	$44.2 \pm 5.0\%$	$51.7 \pm 4.7\%$	$58.6 \pm 3.1\%$	$61.9 \pm 3.7\%$
25	**Clean**	$69.8 \pm 5.0\%$	$78.8 \pm 4.1\%$	$82.7 \pm 3.7\%$	$87.8 \pm 1.5\%$	$88.6 \pm 3.4\%$
	20	$69.7 \pm 6.1\%$	$78.8 \pm 3.9\%$	$82.5 \pm 3.8\%$	$88.1 \pm 1.1\%$	$88.4 \pm 3.4\%$
	10	$70.3 \pm 4.7\%$	$78.6 \pm 4.9\%$	$82.8 \pm 3.7\%$	$85.8 \pm 2.4\%$	$88.3 \pm 4.7\%$
	5	$67.7 \pm 3.7\%$	$78.0 \pm 3.6\%$	$79.8 \pm 2.3\%$	$83.3 \pm 3.0\%$	$86.1 \pm 4.6\%$
	3	$58.4 \pm 3.3\%$	$67.2 \pm 2.9\%$	$69.7 \pm 1.7\%$	$77.7 \pm 2.3\%$	$81.7 \pm 4.3\%$
	0	$38.1 \pm 8.0\%$	$41.1 \pm 5.6\%$	$46.1 \pm 3.0\%$	$52.7 \pm 7.3\%$	$57.5 \pm 1.4\%$
50	**Clean**	$68.9 \pm 4.3\%$	$75.3 \pm 6.7\%$	$80.0 \pm 3.5\%$	$85.0 \pm 3.7\%$	$88.1 \pm 4.9\%$
	20	$69.4 \pm 4.4\%$	$75.3 \pm 6.4\%$	$80.0 \pm 3.7\%$	$85.6 \pm 3.3\%$	$88.1 \pm 5.3\%$
	10	$68.3 \pm 3.7\%$	$75.3 \pm 6.2\%$	$80.6 \pm 3.9\%$	$85.3 \pm 3.9\%$	$87.0 \pm 4.9\%$
	5	$62.7 \pm 4.1\%$	$71.4 \pm 5.3\%$	$76.4 \pm 5.1\%$	$82.5 \pm 4.2\%$	$85.0 \pm 3.9\%$
	3	$51.9 \pm 4.0\%$	$60.6 \pm 4.4\%$	$65.3 \pm 3.1\%$	$72.8 \pm 3.2\%$	$77.8 \pm 3.8\%$
	0	$30.0 \pm 5.7\%$	$36.3 \pm 4.7\%$	$41.3 \pm 5.0\%$	$46.6 \pm 6.2\%$	$52.2 \pm 3.1\%$
75	**Clean**	$54.2 \pm 1.3\%$	$58.3 \pm 5.7\%$	$67.0 \pm 4.8\%$	$74.7 \pm 5.2\%$	$79.2 \pm 6.1\%$
	20	$53.8 \pm 0.9\%$	$58.6 \pm 6.8\%$	$66.9 \pm 4.1\%$	$75.6 \pm 6.3\%$	$79.1 \pm 5.4\%$
	10	$52.3 \pm 2.4\%$	$57.3 \pm 7.4\%$	$65.8 \pm 4.5\%$	$76.1 \pm 7.3\%$	$78.8 \pm 4.2\%$
	5	$47.3 \pm 5.7\%$	$55.5 \pm 3.1\%$	$59.7 \pm 1.1\%$	$69.1 \pm 4.1\%$	$75.8 \pm 3.2\%$
	3	$39.5 \pm 3.7\%$	$45.8 \pm 3.1\%$	$51.9 \pm 0.5\%$	$64.2 \pm 5.5\%$	$68.6 \pm 2.5\%$
	0	$30.6 \pm 2.7\%$	$34.7 \pm 5.0\%$	$40.8 \pm 4.4\%$	$43.9 \pm 0.3\%$	$45.6 \pm 1.4\%$
100	**Clean**	$44.2 \pm 5.7\%$	$56.4 \pm 1.2\%$	$65.3 \pm 5.8\%$	$73.4 \pm 4.7\%$	$79.8 \pm 4.0\%$
	20	$42.8 \pm 7.8\%$	$55.5 \pm 2.4\%$	$65.0 \pm 5.5\%$	$73.0 \pm 3.9\%$	$78.8 \pm 4.7\%$
	10	$40.5 \pm 5.6\%$	$52.2 \pm 3.7\%$	$64.4 \pm 4.8\%$	$73.6 \pm 2.6\%$	$77.3 \pm 3.5\%$
	5	$36.6 \pm 9.8\%$	$48.4 \pm 8.8\%$	$60.9 \pm 8.7\%$	$63.9 \pm 5.6\%$	$72.0 \pm 3.6\%$
	3	$34.2 \pm 7.6\%$	$43.8 \pm 8.3\%$	$54.2 \pm 11.2\%$	$55.6 \pm 3.4\%$	$62.5 \pm 1.4\%$
	0	$26.7 \pm 5.1\%$	$34.4 \pm 4.4\%$	$37.7 \pm 1.1\%$	$40.5 \pm 1.1\%$	$42.8 \pm 2.6\%$

Table 7.2: Classification scores of LRM-NMD in low-resolution multi-label learning mode on the clean data in function of the number of labelled events per annotating segment (n_{lbl}), semantic balancing parameter (η) and sparsity penalisation (λ)

λ	η	**Number of labelled events per annotating segment** (n_{lbl})				
		1	**2**	**3**	**4**	**5**
0	**0**	$34.8 \pm 3.3\%$	$34.8 \pm 3.3\%$	$34.8 \pm 3.3\%$	$34.8 \pm 3.3\%$	$34.8 \pm 3.3\%$
	1	$37.7 \pm 2.7\%$	$42.3 \pm 3.1\%$	$46.3 \pm 2.7\%$	$49.4 \pm 4.5\%$	$51.4 \pm 4.1\%$
	2	$41.1 \pm 1.5\%$	$47.5 \pm 2.3\%$	$53.6 \pm 2.8\%$	$55.0 \pm 3.7\%$	$59.5 \pm 2.7\%$
	3	$42.3 \pm 1.5\%$	$48.4 \pm 1.1\%$	$56.1 \pm 3.5\%$	$59.2 \pm 2.6\%$	$64.7 \pm 3.3\%$
	4	$42.8 \pm 2.3\%$	$50.3 \pm 2.1\%$	$57.8 \pm 4.3\%$	$61.3 \pm 2.1\%$	$66.7 \pm 3.3\%$
	5	$42.7 \pm 2.1\%$	$53.1 \pm 3.1\%$	$58.9 \pm 3.1\%$	$62.7 \pm 2.8\%$	$69.7 \pm 3.4\%$
	10	$45.2 \pm 1.9\%$	$51.7 \pm 6.7\%$	$59.4 \pm 2.0\%$	$65.5 \pm 1.9\%$	$75.2 \pm 3.7\%$
	15	$45.2 \pm 4.9\%$	$47.8 \pm 9.0\%$	$59.5 \pm 4.4\%$	$68.0 \pm 2.0\%$	$77.3 \pm 3.9\%$
	20	$41.7 \pm 3.3\%$	$44.1 \pm 9.5\%$	$59.2 \pm 7.5\%$	$69.7 \pm 2.8\%$	$78.0 \pm 4.8\%$
	25	$38.0 \pm 4.9\%$	$44.1 \pm 8.1\%$	$58.6 \pm 7.3\%$	$70.0 \pm 3.3\%$	$78.3 \pm 4.6\%$
	30	$37.2 \pm 6.1\%$	$44.1 \pm 8.4\%$	$58.4 \pm 7.6\%$	$69.2 \pm 3.1\%$	$78.3 \pm 5.1\%$
5	**0**	$60.0 \pm 3.7\%$	$60.0 \pm 3.7\%$	$60.0 \pm 3.7\%$	$60.0 \pm 3.7\%$	$60.0 \pm 3.7\%$
	1	$66.3 \pm 3.6\%$	$70.6 \pm 4.2\%$	$75.3 \pm 4.9\%$	$81.3 \pm 5.1\%$	$84.7 \pm 3.7\%$
	2	$70.3 \pm 3.4\%$	$76.9 \pm 7.0\%$	$82.3 \pm 3.6\%$	$87.2 \pm 3.1\%$	$89.5 \pm 3.1\%$
	3	$72.8 \pm 4.2\%$	$80.3 \pm 5.4\%$	$85.6 \pm 2.1\%$	$88.6 \pm 2.1\%$	$90.2 \pm 2.9\%$
	4	$73.4 \pm 3.7\%$	$81.6 \pm 4.4\%$	$86.3 \pm 1.5\%$	$\underline{89.5 \pm 1.7\%}$	$90.0 \pm 3.7\%$
	5	$\underline{73.9 \pm 5.3\%}$	$\underline{82.3 \pm 5.6\%}$	$\underline{86.7 \pm 2.1\%}$	$89.1 \pm 1.2\%$	$90.2 \pm 3.9\%$
	10	$73.1 \pm 6.0\%$	$80.6 \pm 1.8\%$	$83.9 \pm 2.5\%$	$87.5 \pm 1.6\%$	$90.6 \pm 3.8\%$
	15	$68.4 \pm 4.9\%$	$75.3 \pm 3.9\%$	$83.3 \pm 1.4\%$	$87.3 \pm 1.1\%$	$90.6 \pm 3.4\%$
	20	$59.7 \pm 4.9\%$	$73.4 \pm 5.4\%$	$82.7 \pm 3.5\%$	$88.1 \pm 1.4\%$	$90.6 \pm 2.9\%$
	25	$57.2 \pm 3.7\%$	$70.3 \pm 7.8\%$	$81.3 \pm 3.1\%$	$86.4 \pm 3.1\%$	$\underline{90.8 \pm 2.7\%}$
	30	$55.8 \pm 5.1\%$	$65.8 \pm 9.6\%$	$79.1 \pm 4.0\%$	$85.2 \pm 4.8\%$	$\underline{90.8 \pm 2.7\%}$
10	**0**	$56.1 \pm 2.2\%$	$56.1 \pm 2.2\%$	$56.1 \pm 2.2\%$	$56.1 \pm 2.2\%$	$56.1 \pm 2.2\%$
	1	$60.3 \pm 3.0\%$	$63.6 \pm 1.3\%$	$68.3 \pm 3.1\%$	$72.5 \pm 5.7\%$	$75.8 \pm 3.9\%$
	2	$63.1 \pm 4.3\%$	$69.2 \pm 4.3\%$	$74.5 \pm 3.9\%$	$81.6 \pm 2.1\%$	$85.3 \pm 5.4\%$
	3	$63.6 \pm 4.6\%$	$72.2 \pm 4.9\%$	$79.4 \pm 3.1\%$	$85.8 \pm 2.9\%$	$86.4 \pm 4.6\%$
	4	$63.4 \pm 4.0\%$	$73.3 \pm 4.0\%$	$81.3 \pm 3.8\%$	$86.3 \pm 2.0\%$	$87.2 \pm 5.1\%$
	5	$64.2 \pm 4.3\%$	$75.5 \pm 4.6\%$	$80.5 \pm 3.1\%$	$86.9 \pm 0.9\%$	$87.5 \pm 4.8\%$
	10	$65.2 \pm 4.3\%$	$77.2 \pm 3.3\%$	$80.5 \pm 3.5\%$	$86.7 \pm 1.3\%$	$88.1 \pm 4.4\%$
	15	$61.3 \pm 6.6\%$	$72.7 \pm 5.8\%$	$80.8 \pm 2.9\%$	$86.6 \pm 1.3\%$	$88.4 \pm 4.3\%$
	20	$53.0 \pm 7.2\%$	$72.0 \pm 4.2\%$	$78.1 \pm 2.7\%$	$83.4 \pm 4.3\%$	$88.4 \pm 4.3\%$
	25	$46.4 \pm 8.3\%$	$68.4 \pm 4.3\%$	$76.3 \pm 5.5\%$	$82.7 \pm 4.0\%$	$88.4 \pm 4.3\%$
	30	$42.3 \pm 10.3\%$	$63.4 \pm 2.8\%$	$74.4 \pm 5.7\%$	$82.4 \pm 4.2\%$	$88.4 \pm 4.3\%$

Note: the best classification scores for each n_{lbl} setting are underlined.

7.3 Relation kernel matrix and feature map

Let us define the $t \times t$ dense kernel matrix $\mathbf{\Omega}_{(tt)}$ by

$$\mathbf{\Omega}_{(tt)} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_t, \mathbf{x}_1) & \cdots & k(\mathbf{x}_t, \mathbf{x}_t) \end{bmatrix}, \tag{7.2}$$

and assume it is of rank m with $m \ll t$. The eigendecomposition of $\mathbf{\Omega}_{(tt)}$ is given by

$$\mathbf{\Omega}_{(tt)} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top, \tag{7.3}$$

with $\mathbf{U}$ the set of $t \times m$ eigenvectors and $\mathbf{\Lambda}$ being the $m \times m$ diagonal matrix of eigenvalues. Let us now relabel $\mathbf{\Omega}_{(tt)}$ to

$$\mathbf{\Omega}_{(tt)} = \begin{bmatrix} \mathbf{\Omega}_{(mm)} & \mathbf{\Omega}_{(jm)}^\top \\ \mathbf{\Omega}_{(jm)} & \mathbf{\Omega}_{(jj)} \end{bmatrix}, \tag{7.4}$$

with $j = t - m$. We can now express the eigendecomposition of $\mathbf{\Omega}_{(tt)}$ by

$$\begin{aligned} \mathbf{\Omega}_{(tt)} &= \begin{bmatrix} \mathbf{U}_{(mm)} \\ \mathbf{U}_{(jm)} \end{bmatrix} \mathbf{\Lambda} \begin{bmatrix} \mathbf{U}_{(mm)} \\ \mathbf{U}_{(jm)} \end{bmatrix}^\top, \\ &= \begin{bmatrix} \mathbf{U}_{(mm)}\mathbf{\Lambda}\mathbf{U}_{(mm)}^\top & \mathbf{U}_{(mm)}\mathbf{\Lambda}\mathbf{U}_{(jm)}^\top \\ \mathbf{U}_{(jm)}\mathbf{\Lambda}\mathbf{U}_{(mm)}^\top & \mathbf{U}_{(jm)}\mathbf{\Lambda}\mathbf{U}_{(jm)}^\top \end{bmatrix}. \end{aligned} \tag{7.5}$$

We can now obtain $\mathbf{U}_{(mm)}$ and $\mathbf{\Lambda}$ by applying the eigendecomposition of $\mathbf{\Omega}_{(mm)} = \mathbf{U}_{(mm)}\mathbf{\Lambda}\mathbf{U}_{(mm)}^\top$. This enables us to determine $\mathbf{U}_{(jm)}$ by computing $\mathbf{U}_{(jm)} = \mathbf{\Omega}_{(jm)}\mathbf{U}_{(mm)}\mathbf{\Lambda}^{-1}$. According to Equation 7.4 and 7.5 it can be shown that $\mathbf{\Omega}_{(jj)} = \mathbf{U}_{(jm)}\mathbf{\Lambda}\mathbf{U}_{(jm)}^\top$. Substituting $\mathbf{U}_{(jm)}$ gives

$$\begin{aligned} \mathbf{\Omega}_{(jj)} &= \mathbf{U}_{(jm)}\mathbf{\Lambda}\mathbf{U}_{(jm)}^\top, \\ &= (\mathbf{\Omega}_{(jm)}\mathbf{U}_{(mm)}\mathbf{\Lambda}^{-1})\mathbf{\Lambda}(\mathbf{\Omega}_{(jm)}\mathbf{U}_{(mm)}\mathbf{\Lambda}^{-1})^\top, \\ &= \mathbf{\Omega}_{(jm)}\mathbf{U}_{(mm)}(\mathbf{\Lambda}^{-1}\mathbf{\Lambda})\mathbf{\Lambda}^{-1}\mathbf{U}_{(mm)}^\top\mathbf{\Omega}_{(jm)}^\top, \\ &= \mathbf{\Omega}_{(jm)}(\mathbf{U}_{(mm)}\mathbf{\Lambda}^{-1}\mathbf{U}_{(mm)}^\top)\mathbf{\Omega}_{(jm)}^\top, \\ &= \mathbf{\Omega}_{(jm)}\mathbf{\Omega}_{(mm)}^{-1}\mathbf{\Omega}_{(jm)}^\top, \\ &= (\mathbf{\Omega}_{(jm)}\mathbf{\Omega}_{(mm)}^{-\frac{1}{2}})(\mathbf{\Omega}_{(jm)}\mathbf{\Omega}_{(mm)}^{-\frac{1}{2}})^\top, \\ &= \mathbf{\Phi}_{(jm)}\mathbf{\Phi}_{(jm)}^\top, \end{aligned} \tag{7.6}$$

and thereby showing that the $j \times m$ feature matrix $\boldsymbol{\Phi}_{(jm)}$ is given by $\boldsymbol{\Omega}_{(jm)}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}}$. By using $\boldsymbol{\Omega}_{(mm)}^{\frac{1}{2}}$ as feature matrix $\boldsymbol{\Phi}_{(mm)}$ for the first $m \times m$ points we get that the feature map $\boldsymbol{\Phi}_{(tm)}$ is defined by

$$\begin{aligned}
\boldsymbol{\Phi}_{(tm)} &= \begin{bmatrix} \boldsymbol{\Phi}_{(mm)} \\ \boldsymbol{\Phi}_{(jm)} \end{bmatrix}, \\
&= \begin{bmatrix} \boldsymbol{\Omega}_{(mm)}^{\frac{1}{2}} \\ \boldsymbol{\Omega}_{(jm)}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}} \end{bmatrix}, \\
&= \begin{bmatrix} \boldsymbol{\Omega}_{(mm)} \\ \boldsymbol{\Omega}_{(jm)} \end{bmatrix} \boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}}, \\
&= \boldsymbol{\Omega}_{(tm)}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}}.
\end{aligned} \tag{7.7}$$

We can now verify that $\boldsymbol{\Phi}_{(tm)}$ corresponds to the correct kernel matrix by

$$\begin{aligned}
\boldsymbol{\Phi}_{(tm)}\boldsymbol{\Phi}_{(tm)}^{\top} &= \begin{bmatrix} \boldsymbol{\Omega}_{(mm)}^{\frac{1}{2}} \\ \boldsymbol{\Omega}_{(jm)}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Omega}_{(mm)}^{\frac{1}{2}} \\ \boldsymbol{\Omega}_{(jm)}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}} \end{bmatrix}^{\top}, \\
&= \begin{bmatrix} \boldsymbol{\Omega}_{(mm)}^{\frac{1}{2}}\boldsymbol{\Omega}_{(mm)}^{\frac{1}{2}} & \boldsymbol{\Omega}_{(mm)}^{\frac{1}{2}}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}}\boldsymbol{\Omega}_{(jm)}^{\top} \\ \boldsymbol{\Omega}_{(jm)}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}}\boldsymbol{\Omega}_{(mm)}^{\frac{1}{2}} & \boldsymbol{\Omega}_{(jm)}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}}\boldsymbol{\Omega}_{(mm)}^{-\frac{1}{2}}\boldsymbol{\Omega}_{(jm)}^{\top} \end{bmatrix}, \\
&= \begin{bmatrix} \boldsymbol{\Omega}_{(mm)} & \boldsymbol{\Omega}_{(jm)}^{\top} \\ \boldsymbol{\Omega}_{(jm)} & \boldsymbol{\Omega}_{(jm)}\boldsymbol{\Omega}_{(mm)}^{-1}\boldsymbol{\Omega}_{(jm)}^{\top} \end{bmatrix}, \\
&= \begin{bmatrix} \boldsymbol{\Omega}_{(mm)} & \boldsymbol{\Omega}_{(jm)}^{\top} \\ \boldsymbol{\Omega}_{(jm)} & \boldsymbol{\Omega}_{(jj)} \end{bmatrix}, \\
&= \boldsymbol{\Omega}_{(tt)}.
\end{aligned} \tag{7.8}$$

7.4 FS-LSSVM normal step updates

Let us define the $m \times m$ cross-product matrix by $\mathbf{P}_{(tm)} = \boldsymbol{\Phi}_{(tm)}^{\top}\boldsymbol{\Phi}_{(tm)} + \frac{1}{\gamma}\mathbf{I}$. The solution of Equation 5.13 becomes $\mathbf{w}_{(tm)} = \mathbf{P}_{(tm)}^{-1}\boldsymbol{\Phi}_{(tm)}^{\top}\mathbf{y}_{(t)}$ with an incurred cost of $\zeta_{(tm)} = J(\mathbf{w}_{(tm)})$. Each time a new sample is presented to the algorithm,

the design matrix and label vector are augmented by

$$\boldsymbol{\Phi}_{(tm)} = \begin{bmatrix} \boldsymbol{\Phi}_{(t-1,m)} \\ \hat{\varphi}(\mathbf{x}_t)^\top \end{bmatrix} \quad \text{and} \quad \mathbf{y}_{(t)} = \begin{bmatrix} \mathbf{y}_{(t-1)} \\ y_t \end{bmatrix}. \tag{7.9}$$

The cross-product matrix can be updated by $\mathbf{P}_{(tm)} = \mathbf{P}_{(t-1,m)} + \hat{\varphi}(\mathbf{x}_t)\hat{\varphi}(\mathbf{x}_t)^\top$ which results in following set of RLS updates:

$$\mathbf{P}^{-1}_{(tm)} = \mathbf{P}^{-1}_{(t-1,m)} - \frac{\mathbf{P}^{-1}_{(t-1,m)}\hat{\varphi}(\mathbf{x}_t)\hat{\varphi}(\mathbf{x}_t)^\top\mathbf{P}^{-1}_{(t-1,m)}}{\Delta}, \tag{7.10}$$

$$\mathbf{w}_{(tm)} = \mathbf{w}_{(t-1,m)} + \frac{\epsilon}{\Delta}\mathbf{P}^{-1}_{(t-1,m)}\hat{\varphi}(\mathbf{x}_t), \tag{7.11}$$

$$\zeta_{(tm)} = \zeta_{(t-1,m)} + \frac{\epsilon^2}{\Delta}, \tag{7.12}$$

with $\Delta = 1 + \hat{\varphi}(\mathbf{x}_t)^\top\mathbf{P}^{-1}_{(t-1,m)}\hat{\varphi}(\mathbf{x}_t)$ and $\epsilon = y_t - \mathbf{w}^\top_{(t-1,m)}\hat{\varphi}(\mathbf{x}_t)$. The overall computational complexity for these updates is again $\mathcal{O}(m^2)$.

Bibliography

[1] Alam, M. R., Reaz, M. B. I., and Ali, M. A. M. A review of smart homes—past, present, and future. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42*, 6 (Nov. 2012), 1190–1203.

[2] AlShu'eili, H., Gupta, G. S., and Mukhopadhyay, S. Voice recognition based wireless home automation system. In *2011 4th International Conference on Mechatronics (ICOM)* (May 2011), pp. 1–6.

[3] Aripin, N. B., and Othman, M. B. Voice control of home appliances using android. In *2014 Electrical Power, Electronics, Communicatons, Control and Informatics Seminar (EECCIS)* (Aug. 2014), pp. 142–146.

[4] Ashton, K. That 'internet of things' thing. *RFiD Journal* (2009), 97–114.

[5] Atlas, L., Ostendorf, M., and Bernard, G. D. Hidden markov models for monitoring machining tool-wear. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings* (June 2000), vol. 6, pp. 3887–3890.

[6] Aurino, F., Folla, M., Gargiulo, F., Moscato, V., Picariello, A., and Sansone, C. One-class svm based approach for detecting anomalous audio events. In *2014 International Conference on Intelligent Networking and Collaborative Systems* (Sept 2014), pp. 145–151.

[7] Bakar, U., Ghayvat, H., Hasanm, S., and Mukhopadhyay, S. *Smart Sensors, Measurement and Instrumentation*. 01 2016, ch. Activity and Anomaly Detection in Smart Home: A Survey, pp. 191–220.

[8] Barker, J., Vincent, E., Ma, N., Christensen, H., and Green, P. The pascal chime speech separation and recognition challenge. *Computer Speech and Language 27*, 3 (February 2013), 621–633.

[9] Benezeth, Y., Laurent, H., Emile, B., and Rosenberger, C. Towards a sensor for detecting human presence and characterizing activity. *Energy and Buildings 43*, 2 (2011), 305 – 314.

[10] Bertrand, A. Applications and trends in wireless acoustic sensor networks: A signal processing perspective. In *2011 18th IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT)* (November 2011), pp. 1–6.

[11] Bianchini, M., and Scarselli, F. On the complexity of neural network classifiers: A comparison between shallow and deep architectures. *IEEE Transactions on Neural Networks and Learning Systems 25*, 8 (Augustus 2014), 1553–1565.

[12] Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[13] Bisot, V., Essid, S., and Richard, G. Overlapping sound event detection with supervised nonnegative matrix factorization. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (March 2017), pp. 31–35.

[14] Bisot, V., Serizel, R., Essid, S., and Richard, G. Acoustic scene classification with matrix factorization for unsupervised feature learning. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (March 2016), pp. 6445–6449.

[15] Bisot, V., Serizel, R., Essid, S., and Richard, G. Nonnegative feature learning methods for acoustic scene classification. In *DCASE 2017 - Workshop on Detection and Classification of Acoustic Scenes and Events* (Munich, Germany, Nov. 2017).

[16] Boser, B. E., Guyon, I. M., and Vapnik, V. N. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory* (New York, NY, USA, 1992), COLT '92, ACM, pp. 144–152.

[17] Brown, R., Ghavami, N., Siddiqui, H.-U.-R., Adjrad, M., Ghavami, M., and Dudley, S. Occupancy based household energy disaggregation using ultra wideband radar and electrical signature profiles. *Energy and Buildings 141* (2017), 134 – 141.

[18] Bucak, S. S., and Gunsel, B. Incremental subspace learning via non-negative matrix factorization. *Pattern Recognition 42*, 5 (2009), 788 – 797.

[19] Burges, C. J. Simplified support vector decision rules. In *Proceedings 13th International Conference on Machine Learning* (1996), Morgan Kaufmann, pp. 71–77.

[20] Caicedo, D., and Pandharipande, A. Ultrasonic array sensor for indoor presence detection. In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)* (Aug. 2012), pp. 175–179.

[21] Cantin, R., Kindinis, A., and Michel, P. New approaches for overcoming the complexity of future buildings impacted by new energy constraints. *Futures 44*, 8 (2012), 735 – 745.

[22] Chalmers, C., Fergus, P., Montañez, C. A. C., Sikdar, S., Ball, F., and Kendall, B. Detecting activities of daily living and routine behaviours in dementia patients living alone using smart meter load disaggregation. *CoRR abs/1903.12080* (April 2019).

[23] Chan, M., Estève, D., Escriba, C., and Campo, E. A review of smart homes—present state and future challenges. *Computer Methods and Programs in Biomedicine 91*, 1 (2008), 55 – 81.

[24] Chang, C.-C., and Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology 2*, 3 (April 2011), 1–27.

[25] Chang, S., Mitsumoto, N., and Burdick, J. W. An algorithm for uwb radar-based human detection. In *2009 IEEE Radar Conference* (May 2009), pp. 1–6.

[26] Chapelle, O., Schlkopf, B., and Zien, A. *Semi-Supervised Learning*, 1st ed. The MIT Press, 2010.

[27] Chen, J., Zhang, J., Kam, A. H., and Shue, L. An automatic acoustic bathroom monitoring system. In *2005 IEEE International Symposium on Circuits and Systems* (May 2005), vol. Vol. 2, pp. 1750–1753.

[28] Chen, S., Epps, J., Ambikairajah, E., and Le, P. N. An investigation of crowd speech for room occupancy estimation. In *INTERSPEECH*.

[29] Chen, S., Fern, A., and Todorovic, S. Person count localization in videos from noisy foreground and detections. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015), pp. 1364–1372.

[30] Chen, T., Lee, T., Chen, Y., Ma, T., Chuang, T., Chou, C., Yang, C., Lin, T., and Chen, L. 1.4µw/channel 16-channel eeg/ecog processor

for smart brain sensor soc. In *2010 Symposium on VLSI Circuits* (June 2010), pp. 21–22.

[31] Chen, Z., Jiang, C., and Xie, L. Building occupancy estimation and detection: A review. *Energy and Buildings 169* (2018), 260 – 270.

[32] Cisco. The zettabyte era: Trends and analysis. Tech. rep., Cisco Visual Networking Index, June 2017.

[33] Cooley, J., and Tukey, J. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation 19*, 90 (1965), 297–301.

[34] Costa, A., Keane, M. M., Torrens, J. I., and Corry, E. Building operation and energy performance: Monitoring, analysis and optimisation toolkit. *Applied Energy 101* (2013), 310 – 316. Sustainable Development of Energy, Water and Environment Systems.

[35] Davis, S. B., and Mermelstein, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *ACOUSTICS, SPEECH AND SIGNAL PROCESSING, IEEE TRANSACTIONS ON* (1980), 357–366.

[36] Dekkers, G., Rosas, F., Lauwereins, S., Rajendran, S., Pollin, S., Vanrumste, B., van Waterschoot, T., Verhelst, M., and Karsmakers, P. A multi-layered energy consumption model for smart wireless acoustic sensor networks. Tech. rep., 12 2018.

[37] Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological) 39*, 1 (1977), 1–38.

[38] Diment, A., Heittola, T., and Virtanen, T. Semi-supervised learning for musical instrument recognition. In *21st European Signal Processing Conference (EUSIPCO 2013)* (September 2013), pp. 1–5.

[39] Dolui, K., and Datta, S. K. Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing. In *2017 Global Internet of Things Summit (GIoTS)* (June 2017), pp. 1–6.

[40] Driesen, J., and Van hamme, H. Modelling vocabulary acquisition, adaptation and generalization in infants using adaptive bayesian plsa. *Neurocomput. 74*, 11 (May 2011), 1874–1882.

[41] Engel, Y., Mannor, S., and Meir, R. The kernel recursive least-squares algorithm. *IEEE Transactions on Signal Processing 52*, 8 (Aug. 2004), 2275–2285.

[42] Erickson, V. L., Achleitner, S., and Cerpa, A. E. Poem: Power-efficient occupancy-based energy management system. In *Proceedings of the 12th International Conference on Information Processing in Sensor Networks* (New York, NY, USA, 2013), IPSN '13, ACM, pp. 203–216.

[43] Erickson, V. L., Carreira-Perpiñán, M. A., and Cerpa, A. E. Occupancy modeling and prediction for building energy management. *ACM Trans. Sen. Netw. 10*, 3 (May 2014), 42:1–42:28.

[44] ESI Audio. *MAYA44 ETX user's guide*, September 2006. Rev. 2.

[45] Espinoza, M., Suykens, J. A. K., and De Moor, B. Fixed-size least squares support vector machines: A large scale application in electrical load forecasting. *Computational Management Science 3*, 2 (Apr. 2006), 113–129.

[46] Ess, A., Leibe, B., Schindler, K., and Van Gool, L. A mobile vision system for robust multi-person tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition* (June 2008), pp. 1–8.

[47] Esteves, R. M., and Rong, C. Social impact of privacy in cloud computing. In *2010 IEEE Second International Conference on Cloud Computing Technology and Science* (Nov. 2010), pp. 593–596.

[48] European Commission. *The 2015 ageing report: economic and budgetary projections for the 28 EU member states (2013-2060)*, vol. 3. Directorate-General for Economic and Financial Affairs, 2015.

[49] European Commission. *The 2018 ageing report: Underlying Assumptions and Projection Methodologies.* Directorate-General for Economic and Financial Affairs, 2018.

[50] Fang, S., Xu, L. D., Zhu, Y., Ahati, J., Pei, H., Yan, J., and Liu, Z. An integrated system for regional environmental monitoring and management based on internet of things. *IEEE Transactions on Industrial Informatics 10*, 2 (May 2014), 1596–1605.

[51] Ferguson, B. G., and Lo, K. W. Acoustic cueing for surveillance and security applications. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* (May 2006), vol. 6201, pp. 62011K–62011K–7.

[52] Fernando, T., Denman, S., Sridharan, S., and Fookes, C. Tracking by prediction: A deep generative model for mutli-person localisation and tracking. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (March 2018), pp. 1122–1132.

[53] Fleury, A., Vacher, M., and Noury, N. Svm-based multimodal classification of activities of daily living in health smart homes: Sensors, algorithms, and first experimental results. *IEEE Transactions on Information Technology in Biomedicine 14*, 2 (March 2010), 274–283.

[54] Gage, S. H., Joo, W., Kasten, E., and Biswas, S. Acoustic observations in agricultural landscapess. In *The Ecology of Agricultural Landscapes: Long-Term Research on the Path to Sustainability* (2015), S. K. Hamilton, J. E. Doll, and G. P. Robertso, Eds., Oxford University Press, New York, USA, pp. 360–377.

[55] Geiger, J. T., Hofmann, M., Schuller, B., and Rigoll, G. Gait-based person identification by spectral, cepstral and energy-related audio features. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (May 2013), pp. 458–462.

[56] Geiger, J. T., Kneissl, M., Schuller, B. W., and Rigoll, G. Acoustic gait-based person identification using hidden markov models. In *Proceedings of the 2014 Workshop on Mapping Personality Traits Challenge and Workshop* (New York, NY, USA, 2014), MAPTRAITS '14, ACM, pp. 25–30.

[57] Gemmeke, J. F., Ons, B., Tessema, N., Van hamme, H., van de Loo, J., De Pauw, G., Daelemans, W., Huyghe, J., Derboven, J., Vuegen, L., Van Den Broeck, B., Karsmakers, P., and Vanrumste, B. Self-taught assistive vocal interfaces: an overview of the ALADIN project. In *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France, August 25-29, 2013* (2013), pp. 2039–2043.

[58] Gemmeke, J. F., Vuegen, L., Karsmakers, P., Vanrumste, B., and Van hamme, H. An exemplar-based NMF approach to audio event detection. In *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics* (October 2013), pp. 1–4.

[59] Gold, B., Morgan, N., and Ellis, D. *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*, 2nd ed. John Wiley & Sons, Inc., New York, NY, USA.

[60] Golub, G., Hansen, P., and O'Leary, D. Tikhonov regularization and total least squares. *SIAM Journal on Matrix Analysis and Applications* (1999), 185–194.

[61] Gopavanitha, K., and Nagaraju, S. A low cost system for real time water quality monitoring and controlling using IoT. In *2017*

International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS) (August 2017), pp. 3227–3229.

[62] Gu, B., Sheng, V. S., Tay, K. Y., Romano, W., and Li, S. Incremental support vector learning for ordinal regression. *IEEE Transactions on Neural Networks and Learning Systems 26*, 7 (July 2015), 1403–1416.

[63] Gualtieri, A. J., and Cromp, R. F. Support vector machines for hyperspectral remote sensing classification. In *Proc.SPIE* (1998), vol. 3584, pp. 221–232.

[64] Guo, X., Tiller, D. K., Henze, G. P., and Waters, C. E. The performance of occupancy-based lighting control systems: A review. *Lighting Research & Technology 42*, 4 (2010), 415–431.

[65] Habets, E. A. P. Room impulse response generator. Tech. rep., Technische Universiteit Eindhoven, 2010.

[66] Hager, W. Updating the inverse of a matrix. *SIAM Review 31*, 2 (1989), 221–239.

[67] Hakkani Tur, D., Riccardi, G., and Gorin, A. Active learning for automatic speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (May 2002), vol. 4, pp. 3904–3907.

[68] Hammoud, A., Deriaz, M., and Konstantas, D. Ultrasense: A self-calibrating ultrasound-based room occupancy sensing system. *Procedia Computer Science 109* (2017), 75 – 83.

[69] Han, W., Chan, C. F., Choy, C. S., and Pun, K. P. An efficient mfcc extraction method in speech recognition. In *2006 IEEE International Symposium on Circuits and Systems* (May 2006), pp. 4 pp.–.

[70] Hopkins, M. B., and Lee, P. Efficient logarithmic conversion on an arm microcontroller for real time acoustic monitoring. In *2017 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)* (May 2017), pp. 1–6.

[71] Hoyer, P. O. Non-negative sparse coding. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing* (2002), pp. 557–565.

[72] Huang, Q. Occupancy-driven energy-efficient buildings using audio processing with background sound cancellation. *Buildings 8* (June 2018).

[73] Huang, Q., Ge, Z., and Lu, C. Occupancy estimation in smart buildings using audio-processing techniques. *CoRR abs/1602.08507* (2016).

[74] Ishwarappa, and Anuradha, J. A brief introduction on big data 5vs characteristics and hadoop technology. *Procedia Computer Science 48* (2015), 319–324. International Conference on Computer, Communication and Convergence (ICCC 2015).

[75] Iyengar, S. G., Varshney, P. K., and Damarla, T. On the detection of footsteps based on acoustic and seismic sensing. In *2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers* (Nov. 2007), pp. 2248–2252.

[76] Janvier, M., Alameda-Pineda, X., Girin, L., and Horaud, R. Sound-event recognition with a companion humanoid. In *Humanoids 2012 - IEEE International Conference on Humanoid Robotics* (Osaka, Japan, November 2012), IEEE, pp. 104–111.

[77] Janvier, M., Alameda-Pineda, X., Girin, L., and Horaud, R. Sound representation and classification benchmark for domestic robots. In *2014 IEEE International Conference on Robotics and Automation (ICRA 2014)* (Hong-Kong, China, May 2014), IEEE, pp. 104–111.

[78] Jia, X., Feng, Q., Fan, T., and Lei, Q. Rfid technology and its applications in internet of things (iot). In *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)* (April 2012), pp. 1282–1285.

[79] Jung, T., and Polani, D. Sequential learning with ls-svm for large-scale data sets. In *Artificial Neural Networks – ICANN 2006* (Sept. 2006), vol. 4132, pp. 381–390.

[80] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. Large-scale video classification with convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition* (June 2014), pp. 1725–1732.

[81] Karsmakers, P., Pelckmans, K., Suykens, J. A. K., and Van hamme, H. Fixed-size kernel logistic regression for phoneme classification. In *INTERSPEECH* (2007), ISCA, pp. 78–81.

[82] Karsmakers, P., Van hamme, H., and Suykens, J. *Sparse kernel-based models for speech recognition.* Phd dissertation, KU Leuven, 2010.

[83] Katz, S. Assessing self-maintenance: activities of daily living, mobility, and instrumental activities of daily living. *Journal of the American Geriatrics Society 31*, 12 (December 1983), 721–727.

[84] Khan, A., Nicholson, J., Mellor, S., Jackson, D., Ladha, K., Ladha, C., Hand, J., Clarke, J., Olivier, P., and Plötz, T. Occupancy monitoring using environmental and context sensors and a hierarchical analysis framework. In *BuildSys@SenSys* (2014).

[85] Kinnunen, T., and Li, H. An overview of text-independent speaker recognition: From features to supervectors. *Speech Communication 52*, 1 (2010), 12–40.

[86] Kleiminger, W., Beckel, C., Staake, T., and Santini, S. Occupancy detection from electricity consumption data. pp. 1–8.

[87] Knowels Electronics. *SiSonic Design Guide*, August 2011. Rev. 3.

[88] Knowels Electronics. *SPU0410LR5H datasheet*, March 2013. Rev. H.

[89] Knowels Electronics. *SPH1611LR5H datasheet*, March 2015. Rev. A.

[90] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[91] Labeodan, T., Zeiler, W., Boxem, G., and Zhao, Y. Occupancy measurement in commercial office buildings for demand-driven control applications—a survey and detection system evaluation. *Energy and Buildings 93* (2015), 303 – 314.

[92] Lai, L., Suda, N., and Chandra, V. CMSIS-NN: Efficient neural network kernels for arm cortex-m CPUs. *CoRR*, abs/1801.06601 (Aug. 2018).

[93] Lam, K. P., Höynck, M., Dong, B., Andrews, B., shang Chiou, Y., Benitez, D., and Choi, J. Occupancy detection through an extensive environmental sensor network in an open-plan office building. In *Proc. of Building Simulation 09, an IBPSA Conference.*

[94] Laney, D. 3D data management: Controlling data volume, velocity, and variety. Tech. rep., META Group, February 2001.

[95] Langville, A. N., Meyer, C. D., and Albright, R. Initializations for the nonnegative matrix factorization, 2006.

[96] Laput, G., Zhang, Y., and Harrison, C. Synthetic sensors: Towards general-purpose sensing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2017), CHI '17, ACM, pp. 3986–3999.

[97] Lazarescu, M. T. Design of a wsn platform for long-term environmental monitoring for iot applications. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems 3*, 1 (March 2013), 45–54.

[98] Le Roux, J., Weninger, F. J., and Hershey, J. R. Sparse nmf - half-baked or well done? Tech. Rep. TR2015-023, Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA, March 2015.

[99] LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. 436–444.

[100] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE* (1998), pp. 2278–2324.

[101] Lee, D. D., and Seung, H. S. Learning the parts of objects by non-negative matrix factorization. *Nature 401* (1999), 788–791.

[102] Lee, D. D., and Seung, H. S. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems (NIPS)* (2000), T. K. Leen, T. G. Dietterich, and V. Tresp, Eds., vol. 4, MIT Press, pp. 556–562.

[103] Lee, K. H., and Verma, N. A low-power processor with configurable embedded machine-learning accelerators for high-order and adaptive analysis of medical-sensor signals. *IEEE Journal of Solid-State Circuits 48*, 7 (July 2013), 1625–1637.

[104] Li, R., and He, D. Rotational machine health monitoring and fault detection using emd-based acoustic emission feature quantification. *IEEE Transactions on Instrumentation and Measurement 61*, 4 (April 2012), 990–1001.

[105] Liang, N., Huang, G., Saratchandran, P., and Sundararajan, N. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on Neural Networks 17*, 6 (Nov. 2006), 1411–1423.

[106] Libelium. 50 sensor applications for a smarter world., 2018.

[107] Liu, D., Guan, X., Du, Y., and Zhao, Q. Measuring indoor occupancy in intelligent buildings using the fusion of vision sensors. *Measurement Science and Technology 24* (Jun. 2013), 074023.

[108] Logan, B. Mel frequency cepstral coefficients for music modeling. In *In International Symposium on Music Information Retrieval* (2000).

[109] Maas, A. L., Hannun, A. Y., and Ng, A. Y. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing* (2013).

[110] McKinsey and Company. The internet of things: How to capture the value of IoT, May 2018.

[111] Merilampi, S., and Sirkka, A. *Introduction to Smart eHealth and eCare Technologies*. Devices, Circuits, and Systems. CRC Press, 2016.

[112] Meyn, S., Surana, A., Lin, Y., Oggianu, S. M., Narayanan, S., and Frewen, T. A. A sensor-utility-network method for estimation of occupancy in buildings. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference* (Dec. 2009), pp. 1494–1500.

[113] Misbahuddin, S., Zubairi, J. A., Saggaf, A., Basuni, J., A-Wadany, S., and Al-Sofi, A. IoT based dynamic road traffic management for smart cities. In *2015 12th International Conference on High-capacity Optical Networks and Enabling/Emerging Technologies (HONET)* (December 2015), pp. 1–5.

[114] Mokhtari, G., Zhang, Q., Nourbakhsh, G., Ball, S., and Karunanithi, M. Bluesound: A new resident identification sensor—using ultrasound array and ble technology for smart home platform. *IEEE Sensors Journal 17*, 5 (March 2017), 1503–1512.

[115] Moon, T., and Stirling, W. *Mathematical Methods and Algorithms for Signal Processing*. January 2000.

[116] Moreno, P. J., and Agarwal, S. An experimental study of em-based algorithms for semi-supervised learning in audio classification. In *International Conference on Machine Learning (ICML) Workshop on the Continuum from Labeled to Unlabeled data* (August 2003).

[117] Mun Sim, J., Lee, Y., and Kwon, O. Acoustic sensor based recognition of human activity in everyday life for smart home services. *International Journal of Distributed Sensor Networks 2015* (Sept. 2015), 1–11.

[118] Murthy, A., Han, D., Jiang, D., and Oliveira, T. Lighting-enabled smart city applications and ecosystems based on the IoT. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)* (December 2015), pp. 757–763.

[119] Nandwana, M. K., Ziaei, A., and Hansen, J. H. L. Robust unsupervised detection of human screams in noisy acoustic environments. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (April 2015), pp. 161–165.

[120] Navarro, J., Vidaña-Vila, E., Alsina-Pagès, R. M., and Hervás, M. Real-time distributed architecture for remote acoustic elderly monitoring in residential-scale ambient assisted living scenarios. *Sensors 18*, 8.

[121] Nguyen, T. A., and Aiello, M. Energy intelligent buildings based on user activity: A survey. *Energy and Buildings 56* (2013), 244 – 257.

[122] Ntalampiras, S., Potamitis, I., and Fakotakis, N. On acoustic surveillance of hazardous situations. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing* (April 2009), pp. 165–168.

[123] Ntalampiras, S., Potamitis, I., and Fakotakis, N. Acoustic detection of human activities in natural environments. *AES: Journal of the Audio Engineering Society 60* (September 2012), 686–695.

[124] NXP Semiconductors. *ARM cortex-M7 Processor - Technical Reference Manual*, 2014. Rev. 0.

[125] O'Grady, P. D., and Pearlmutter, B. A. Discovering convolutive speech phones using sparseness and non-negativity constraints. In *Proceedings of the Seventh International Conference on Independent Component Analysis* (2007), pp. 520–527.

[126] Osuna, E. E., and Girosi, F. Reducing the run-time complexity in support vector machines. *Advances in Kernel Methods* (1999), 271–283.

[127] Pan, S. J., Tsang, I. W., Kwok, J. T., and Yang, Q. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks 22*, 2 (Feb. 2011), 199–210.

[128] Pan, S. J., and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering 22*, 10 (Oct. 2010), 1345–1359.

[129] Plamondon, R., and Srihari, S. N. Online and off-line handwriting recognition: a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 1 (Jan. 2000), 63–84.

[130] Plumbley, M. D., Kroos, C., Bello, J. P., Richard, G., Ellis, D. P., and Mesaros, A. *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*. Tampere University of Technology. Laboratory of Signal Processing, 2018.

[131] Popescu, M., Li, Y., Skubic, M., and Rantz, M. An acoustic fall detector system that uses sound height information to reduce the false alarm rate. In *2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (Aug 2008), pp. 4628–4631.

[132] Popescu, M., and Mahnot, A. Acoustic fall detection using one-class classifiers. In *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (Sept 2009), pp. 3505–3508.

[133] Pérez-Lombard, L., Ortiz, J., and Pout, C. A review on buildings energy consumption information. *Energy and Buildings 40*, 3 (2008), 394 – 398.

[134] Pétex, J.-F., Cheraly, A., Rabaste, O., Phan, H.-K., Poullin, D., and Koeniguer, E. Around-the-corner radar: Detection of a human being in non-line of sight. *IET Radar, Sonar and Navigation 9* (Feb. 2015).

[135] Renkens, V., and Van hamme, H. Automatic relevance determination for nonnegative dictionary learning in the gamma-poisson model. *Signal Processing 132* (2017), 121–133.

[136] Reynolds, D. A., and Rose, R. C. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing 3*, 1 (January 1995), 72–83.

[137] Riccardi, G., and Hakkani Tur, D. Active learning: Theory and applications to automatic speech recognition. *IEEE Transactions on Speech and Audio Processing 13*, 4 (July 2005), 504–511.

[138] Robinson, A. J. An application of recurrent nets to phone probability estimation. *Trans. Neur. Netw. 5*, 2 (March 1994), 298–305.

[139] Roma, G., Janer, J., Kersten, S., Schirosa, M., Herrera, P., and Serra, X. Ecological acoustics perspective for content-based retrieval of environmental sounds. *EURASIP Journal on Audio, Speech, and Music Processing 2010*, 1 (2010), 960863.

[140] Roma, G., Nogueira, W., and Herrera, P. Recurrence quantification analysis features for environmental sound recognition. In *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics* (October 2013), pp. 1–4.

[141] Ryoo, M. S., and Aggarwal, J. K. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *2009 IEEE 12th International Conference on Computer Vision (ICCV)* (Los Alamitos, CA, USA, Oct. 2009), IEEE Computer Society, pp. 1593–1600.

[142] S, A., N, S. M., S, A., Natarajan, K., Shobha, K. R., and Paventhan, A. An IoT based 6LoWPAN enabled experiment for water management. In *2015 IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS)* (December 2015), pp. 1–6.

[143] Sabatier, J. M., and Ekimov, A. E. A review of human signatures in urban environments using seismic and acoustic methods. In *2008 IEEE Conference on Technologies for Homeland Security* (May 2008), pp. 215–220.

[144] Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Networks 61* (2015), 85 – 117.

[145] Schroeder, J., Brandes, M., Hollosi, D., Wellmann, J., Wittorf, M., Jung, O., Grützmacher, V., and Goetze, S. Foreign object detection in tires by acoustic event detection. In *Deutsche Jahrestagung für Akustik (DAGA)* (Nürnberg, March 2015), vol. 41, pp. 1266–1269.

[146] Schroeder, J., Wabnik, S., van Hengel, P. W. J., and Goetze, S. *Ambient Assisted Living*, vol. 4. Springer, Berlin, Heidelberg, January 2011, ch. Detection and Classification of Acoustic Events for In-Home Care, pp. 181–195.

[147] Sharma, A. S., and Bhalley, R. Asr — a real-time speech recognition on portable devices. In *2016 2nd International Conference on Advances in Computing, Communication, Automation (ICACCA)* (Sep. 2016), pp. 1–4.

[148] Shwartz-Ziv, R., and Tishby, N. Opening the black box of deep neural networks via information. *CoRR abs/1703.00810* (Aug. 2017).

[149] Smaragdis, P. Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs. In *International Conference on Independent Component Analysis and Signal Separation* (2004), Springer, pp. 494–499.

[150] Smola, A. J., and Schölkopf, B. Sparse greedy matrix approximation for machine learning. Morgan Kaufmann, pp. 911–918.

[151] Soda, S., Nakamura, M., Matsumoto, S., Izumi, S., Kawaguchi, H., and Yoshimoto, M. Implementing virtual agent as an interface for smart home voice control. In *2012 19th Asia-Pacific Software Engineering Conference* (Dec. 2012), pp. 342–345.

[152] Srinivasan, V., Stankovic, J., and Whitehouse, K. Using height sensors for biometric identification in multi-resident homes. In *Pervasive Computing* (Berlin, Heidelberg, 2010), Springer Berlin Heidelberg, pp. 337–354.

[153] Statistica: The Statistics Portal. Average costs of industrial internet of things (IoT) sensors from 2004 to 2020 (in U.S. dollars), 2018.

[154] Statistica: The Statistics Portal. Internet of things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions), 2018.

[155] Stebbins, W. C. *The acoustic sense of animals.* Harvard University Press, 1983.

[156] Steckel, J., and Peremans, H. Ultrasound-based air leak detection using a random microphone array and sparse representations. In *SENSORS, 2014 IEEE* (Nov 2014), pp. 1026–1029.

[157] Stevens, S. S., Volkmann, J., and Newman, E. B. A scale for the measurement of the psychological magnitude pitch. *Acoustical Society of America 8* (January 1937), 185–190.

[158] Stojkoska, B. L. R., and Trivodaliev, K. V. A review of internet of things for smart home: Challenges and solutions. *Journal of Cleaner Production 140* (2017), 1454 – 1464.

[159] Stowell, D., Giannoulis, D., Benetos, E., Lagrange, M., and Plumbley, M. D. Detection and classification of acoustic scenes and events. *IEEE Transactions on Multimedia 17*, 10 (October 2015), 1733–1746.

[160] Stowell, D., Giannoulis, D., Benetos, E., Lagrange, M., and Plumbley, M. D. Detection and classification of acoustic scenes and events. *IEEE Transactions on Multimedia 17*, 10 (October 2015), 1733–1746.

[161] Suykens, J., and Vandewalle, J. Least squares support vector machine classifiers. *Neural Processing Letters 9*, 3 (Jun. 1999), 293–300.

[162] Sze, V., Chen, Y., Emer, J. S., Suleiman, A., and Zhang, Z. Hardware for machine learning: Challenges and opportunities. *CoRR abs/1612.07625* (2016).

[163] Sze, V., Chen, Y., Yang, T., and Emer, J. S. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE 105*, 12 (December 2017), 2295–2329.

[164] Taleb, T., Samdanis, K., Mada, B., Flinck, H., Dutta, S., and Sabella, D. On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration. *IEEE Communications Surveys Tutorials 19*, 3 (2017), 1657–1681.

[165] Tan, V. Y. F., and Févotte, C. Automatic relevance determination in nonnegative matrix factorization. In *SPARS'09 - Signal Processing with Adaptive Sparse Structured Representations* (Saint Malo, United Kingdom, Apr 2009), R. Gribonval, Ed., Inria Rennes - Bretagne Atlantique.

[166] Tang, X., Liu, Y., Zheng, L., Ma, C., and Wang, H. Leak detection of water pipeline using wavelet transform method. In *2009 International Conference on Environmental Science and Information Application Technology* (July 2009), vol. 2, pp. 217–220.

[167] Tarzia, S. P., Dick, R. P., Dinda, P. A., and Memik, G. Sonar-based measurement of user presence and attention. In *Proceedings of the 11th International Conference on Ubiquitous Computing* (New York, NY, USA, 2009), UbiComp '09, ACM, pp. 89–92.

[168] Taylor, F. *Digital Filters: Principles and Applications with MATLAB*. Wiley-IEEE Press, Sept. 2011.

[169] Temko, A., Malkin, R., Zieger, C., Macho, D., and Nadeu, C. Acoustic event detection and classification in smart-room environments: Evaluation of chil project systems. *Cough 65*, 48 (2006), 5.

[170] Texas Instruments. *TLV320ADC3101 datasheet*, November 2008. Revision 2018.

[171] Thakur, T. T., Naik, A., Vatari, S., and Gogate, M. Real time traffic management using internet of things. In *2016 International Conference on Communication and Signal Processing (ICCSP)* (April 2016), pp. 1950–1953.

[172] Tripathy, A. K., Mishra, A. K., and Das, T. K. Smart lighting: Intelligent and weather adaptive lighting in street lights using IoT. In *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)* (July 2017), pp. 1236–1239.

[173] Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. *The development of an acoustic based home monitoring system to improve home care*. Phd dissertation, KU Leuven, 2017.

[174] Van Den Broeck, B., Vuegen, L., Van hamme, H., Moonen, M., Karsmakers, P., and Vanrumste, B. Footstep localization based on in-home microphone-array signals. In *In proceedings of Assistive technology: From research to practice* (Sept. 2013), vol. 33, pp. 90–94.

[175] Verma, N., Shoeb, A., Guttag, J. V., and Chandrakasan, A. P. A micro-power eeg acquisition soc with integrated seizure detection processor for continuous patient monitoring. In *2009 Symposium on VLSI Circuits* (June 2009), pp. 62–63.

[176] Virtanen, T., Mesaros, A., Heittola, T., Plumbley, M., Foster, P., Benetos, E., and Lagrange, M. *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2016 Workshop (DCASE2016)*. Tampere University of Technology. Department of Signal Processing, 2016.

[177] Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. Automatic monitoring of activities of daily living based on real-life acoustic sensor data: A preliminary study. In *Fourth workshop on speech and language processing for assistive technologies (SLPAT)* (August 2013), pp. 113–118.

[178] Wang, D., and Brown, G. J. *Computational auditory scene analysis: principles, algorithms, and applications*. Wiley-IEEE Press, October 2006.

[179] Wang, H., Raj, B., and Xing, E. P. On the origin of deep learning. *CoRR abs/1702.07800* (2017).

[180] Wang, T., Gong, S., Zhu, X., and Wang, S. Person re-identification by video ranking. In *Computer Vision – ECCV 2014* (Cham, 2014), D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Springer International Publishing, pp. 688–703.

[181] Williams, C. K. I., and Seeger, M. Using the nystrom method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13* (2001), T. K. Leen, T. G. Dietterich, and V. Tresp, Eds., MIT Press, pp. 682–688.

[182] Yavari, E., Jou, H., Lubecke, V., and Boric-Lubecke, O. Doppler radar sensor for occupancy monitoring. In *2013 IEEE Topical Conference on Power Amplifiers for Wireless and Radio Applications* (Jan. 2013), pp. 145–147.

[183] Young, S. J., Kershaw, D., Odell, J., Ollason, D., Valtchev, V., and Woodland, P. *The HTK Book Version 3.4*. Cambridge University Press, 2006.

[184] Zappi, P., Farella, E., and Benini, L. Enhancing the spatial resolution of presence detection in a pir based wireless surveillance network. In *2007 IEEE Conference on Advanced Video and Signal Based Surveillance* (Sep. 2007), pp. 295–300.

[185] Zhong, Z., Suzuki, J., Miyake, T., Kondou, H., and Enastu, K. Study on water leak detection using wavelet instantaneous cross-correlation. In *2015 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR)* (July 2015), pp. 133–137.

[186] Zhou, G., Yang, Z., Xie, S., and Yang, J. M. Online blind source separation using incremental nonnegative matrix factorization with volume constraint. *IEEE Transactions on Neural Networks 22*, 4 (April 2011), 550–560.

[187] Zhou, Z.-H. A brief introduction to weakly supervised learning. *National Science Review 5*, 1 (2018), 44–53.

List of publications

Papers in international journals

- Vuegen, L., Karsmakers, P., Vanrumste, B., and Van hamme, H. "Acoustic event classification using low-resolution multi-label non-negative matrix deconvolution." In Journal of Audio Engineering Society, AES vol. 66, nr. 5, May 2018, pp. 369-384.
  (**Awarded as the best JAES paper of 2018**)

Papers in proceedings of international conferences

- Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Energy efficient monitoring of activities of daily living using a wireless acoustic sensor network in clean and noisy conditions." In proceedings of 23rd European Signal Processing Conference, EUSIPCO 2015, Nice, France, Sept. 2015, pp. 449-453.
- Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Monitoring activities of daily living using wireless acoustic sensor networks in clean and noisy conditions." In proceedings of 37th international conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2015, Milan, Italy, Aug. 2015, pp. 4966-4969.
- Gemmeke, J. F., Vuegen, L., Karsmakers, P., Vanrumste, B., and Van hamme, H. "An exemplar-based NMF approach to audio event detection." In proceedings of IEEE workshop on applications of signal processing to audio and acoustics , WASPAA 2013, New Paltz, New York, USA, Okt. 2013, pp. 1-4.
- Van Den Broeck, B., Vuegen, L., Van hamme, H., Moonen, M., Karsmakers, P., and Vanrumste, B. "Footstep localization based on in-

home microphone-array signals." In proceedings of Assistive technology: From research to practice, AAATE 2013, Vilamoura, Portugal, Sept. 2013, pp. 90-94.

- Gemmeke, J. F., Ons, B., Tessema, N., Van hamme, H., van de Loot, J., De Pauw, G., Daelemans, W., Huyghe, J., Derboven, J., Vuegen, L., Van Den Broeck, B., Karsmakers, P., Vanrumste, B. "Self-taught assistive vocal interfaces: An overview of the ALADIN project." In proceedings of 14th Annual Conference of the International-Speech-Communication-Association, INTERSPEECH 2013, Lyon, France, Aug. 2013, pp. 2038-2042.

- Vuegen, L., Van Den Broeck, B., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Automatic monitoring of activities of daily living based on real-life acoustic sensor data: A preliminary study." In proceedings of 4th workshop on speech and language processing for assistive technologies, SLPAT 2013, Grenoble, France, Aug. 2013, pp. 113-118.

- Meire, M., Vuegen, L., and Karsmakers, P. "The impact of missing labels and overlapping sound events on multi-label multi-instance learning for sound event classification." Accepted for the proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events, DCASE 2019, New York, United-States, Oct. 2019.

Abstracts in proceedings of international conferences

- Vuegen, L., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Weakly-supervised classification of domestic acoustic events for indoor monitoring applications." In proceedings of IEEE Conference on Biomedical and Health Informatics 2018, BHI 2018, Las Vegas, Nevada, USA, March 2018.

- Vuegen, L., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Automatic monitoring of activities of daily living using a wireless acoustic sensor network." In proceedings of 5th Dutch Bio-Medical Engineering Conference 2015, BME 2015, Egmond aan Zee, The Netherlands, Jan. 2015.

- Van Den Broeck, B., Vuegen, L., Dekkers, G., Van hamme, H., Moonen, M., Vanrumste, B. and Karsmakers, P. "Footstep localizating by acoustic signals." In proceedings of Belgian Day on Biomedical Engineering, Brussels, Belgium, Dec. 2013.

- Vuegen, L., Van Den Broeck, B., Karsmakers, P., Gemmeke, J. F., Vanrumste, B., and Van hamme, H. "An MFCC-GMM approach for event detection and classification." In proceedings of IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events, DCASE 2013, New Paltz, New York, USA, Okt. 2013.
- Vuegen, L., Karsmakers, P., Van hamme, H., and Vanrumste, B. "Ultrasound Based Automatic home observation." In proceedings of 4th Dutch Bio-Medical Engineering Conference 2013, BME 2013, Egmond aan Zee, The Netherlands, Jan. 2013.

Book chapters

- Luca, S., Vuegen, L., Van hamme, H., Karsmakers, P., and Vanrumste, B. "Decision support systems for home monitoring applications: Classification of activities of daily living and epileptic seizures.", Bookchapter: In Institution of Engineering and Technology, IET 2016, London, UK, pp. 271-291.

Scientific reports

- Dekkers, G., Vuegen, L., van Waterschoot, T., Vanrumste, B., and Karsmakers, P., "DCASE 2018 Challenge - Task 5: Monitoring of domestic activities based on multi-channel acoustics.", DCASE 2018, Surrey, London, UK, Nov. 2018.

Internal written reports

- Vuegen, L. and Karsmakers, P. "On-edge online acoustic presence detection." In final VLAIO report for the O&O project: Research towards a perceptive sensing home control system (PERSES).

Biography

Lode Vuegen was born on 2 June 1989 in Neerpelt, Belgium. He received his Master of Science degree in Applied Engineering Technology ('industrieel ingenieur') option Electronics-ICT from the Katholieke Hogeschool Kempen, Geel, Belgium, in 2012. His PhD research is carried out at the KU Leuven ADVISE research group. It was funded by a VLAIO doctoral scholarship (contract 121565) and a VLAIO O&O project named PERSES.



FACULTY OF ENGINEERING TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING
ADVISE
Kleinhoefstraat 4
B-2440 Geel
lode.vuegen@kuleuven.be
http://www.kuleuven.be/advise